

UNCLASSIFIED

AD NUMBER
AD476557
NEW LIMITATION CHANGE
TO Approved for public release, distribution unlimited
FROM Distribution authorized to U.S. Gov't. agencies and their contractors; Specific Authority, Dec 1965. Other requests shall be referred to Rome Air Development Center, Griffis AFB, NY.
AUTHORITY
RADC USAF ltr, 22 Feb 1971

THIS PAGE IS UNCLASSIFIED

476557
RADC-TR-65-415
Final Report



SECURITY TECHNIQUES FOR EDP OF
MULTILEVEL CLASSIFIED INFORMATION

Harvey W. Bingham

TECHNICAL REPORT NO. RADC-TR- 65-415
December 1965

Information Processing Branch
Rome Air Development Center
Research and Technology Division
Air Force Systems Command
Griffiss Air Force Base, New York

This document is subject to special
export controls and each transmittal
to foreign governments or foreign
nationals may be made only with
prior approval of RADC (EMLI),
GAFB, N.Y.

Best Available Copy

JAN 22 1966
RADC-TR-65-415
RADC-TR-65-415

When US Government drawings, specifications, or other data are used for any purpose other than a definitely related government procurement operation, the government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data is not to be regarded, by implication or otherwise, as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacturer, use, or sell any patented invention that may in any way be related thereto.

Foreign announcement and distribution of this report is not authorized.

Best Available Copy

Do not return this copy. Retain or destroy.

**SECURITY TECHNIQUES FOR EDP OF
MULTILEVEL CLASSIFIED INFORMATION**

Harvey W. Bingham

**This document is subject to special
export controls and each transmittal
to foreign governments or foreign
nationals may be made only with
prior approval of RADC (EMLI),
GAFB, N.Y.**


FOREWORD

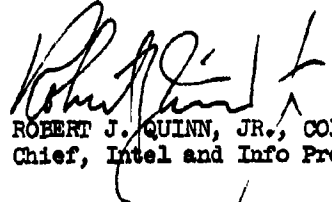
In February 1965, Burroughs Defense, Space and Special Systems Group undertook an eight-month study program for Rome Air Development Center. The area of interest fell within the Air Force System Command's Project No. 4594, Non-Numerical Data Processing for Intelligence, Task No. 459404. This work was performed under Air Force Contract No. AF30(602)-3596. The report is identified by the contractor as Burroughs 4424-65-112.

The Rome Air Development Project Engineer was Major Louis A. Roberts, Jr. (EMIIO).

Release of subject report to the general public is prohibited by the Strategic Trade Control Program, Mutual Defense Assistance Control List (revised 15 June 1964), published by the Department of State.

This technical report has been reviewed and is approved.


Approved: LOUIS A. ROBERTS, JR., MAJOR, USAF
Project Engineer


Approved: ROBERT J. QUINN, JR., COLONEL, USAF
Chief, Intel and Info Processing Div

FOR THE COMMANDER:


IRVING J. CABELMAN
Chief, Advanced Studies Group

ABSTRACT

The study objective was to develop hardware and software techniques for security (need-to-know) control of on-line users and programmers in multiprogramming, multiprocessing EDP systems of apparent future development. Hardware techniques recommended include: (1) processors having two modes of operation, interrupt entry into control mode in which privileged instructions are executable, flag bits for identification and control of memory words, and address checks against access-differentiated memory bounds; (2) parity checks on intermodule information transfers; (3) input/output control processors which establish and verify peripheral unit connections, check memory addresses against bounds, and confirm security content of record headers being transferred; and (4) bulk file control of physical record integrity, and lock control over write permission and flag bit setting to permit supervisor establishment of control programs. Software techniques reside in the executive control program and are executed in control mode and identified by flag bits. Security routines are described and evaluated which construct, protect, and check access requests against user security control profiles, verify memory bounds and memory blanking, and provide security indicators for input/output. The integrated techniques are applied to control users and system programmers in an advanced modular system. Retrofit of most of the recommended techniques to an existing data processor (the Burroughs D825 modular data processing system) is feasible. An external retrofit unit is described which provides control mode and privileged instructions for single-mode processors

CONTENTS

	<u>Page</u>
ABSTRACT	iii
GLOSSARY	ix
Section I, SUMMARY ANALYSIS AND IMPLICATIONS FOR FUTURE PLANNING	1
Background of the Program	1
Statement of Objectives	2
Study Constraints	4
Postulates and Their Security Implications	5
Application of Selected Security Processing Techniques to a Hypothetical Case	7
Implications for Future Planning.	14
Section II, PHYSICAL SYSTEM CONFIGURATIONS	17
Processing Systems Analyzed	17
Information Flow Example.	24
Security Problem Areas	25
Section III, HARDWARE SECURITY TECHNIQUES	31
Processor	32
Processor Modes.	33
Interrupts	35
Flag Bits	36
Privileged Instructions	38
Memory Bounds Registers.	40
Processor Hardware Access Restriction	44
High-Speed Memory Module	44
Input/Output Control Processor	45
Work Station	51
Shared Use of a Work Station.	52
Bulk File Controller.	53

CONTENTS (Cont'd)

	<u>Page</u>
Bulk File	54
Hardware Additions for Security Protection	55
Section IV, SOFTWARE SECURITY TECHNIQUES	59
Operating System and Executive Control Program	59
Tables and Directories	60
Special Tables	52
Equivalence Preserving Transformation	64
Redundant Programming	65
Control of User Production Jobs	67
Control of User Programming Actions	69
System Activity Logging	70
Section V, ECP APPLICATION TO CONTROL USER PROCESSING	73
Startup	77
Loading ECP	77
Establishing the Available Equipment Configuration	78
Forming System User Table	79
Releasing the System Under ECP Control	80
Using the System	80
Initial User Identification and Authentication	85
User Interrupt to Enter Processing Input	86
Processing User Program	88
Input/Output Processing	89
Overlay Control	94
Safe_guards on the ECP	95
ECP Alteration by System Programming	95
ECP Self-protection	96
Control of User's Control Profiles	97
Summary of Software Recommended for Security Control Application	98
Section VI, IOCP APPLICATION FOR SECURITY CONTROL	101
Flow Through IOCP From Bulk File	101
Output Flow Through IOCP to Bulk File	104
Input Flow Through IOCP From Terminal Unit	107
Output Flow Through IOCP to Terminal Unit	109
Section VII, SECURITY TECHNIQUE RETROFIT APPLICATION	113
Computer Module	113
Memory Module	116
Input/Output Module	116
Switching Exchanges	117

CONTENTS (Cont'd)

	<u>Page</u>
Section VIII, EVALUATION OF ALL SECURITY TECHNIQUES CONSIDERED DURING THE PROGRAM	119
Section IX, BIBLIOGRAPHY	129
 APPENDICES	
I Security Routines	135
II Alarm Associated With Privileged Instructions	151
III Memory Protection	157
IV Achieving Reliable Machine Operations	165
V Work Station With Multiple Simultaneous Users	169
VI Study Contributors	173

ILLUSTRATIONS

<u>Figure</u>		
1.	Interrelationships of the Recommended Security Control Techniques	8
2.	Block Diagram for Future Multiprogramming, Multiprocessing EDP Systems	20
3.	Processor Module	33
4.	I/O Control Processor	47
5.	Bulk File Controller.	53
6.	User Job Flow Control by ECP	75
7.	Input Flow Through IOCP From Bulk File	102
8.	Output Flow Through IOCP to Bulk File	105
9.	Input Flow Through IOCP From Terminal Unit	108
10.	Output Flow Through IOCP to Terminal Unit.	110
11.	Fail-Safe Privileged Instruction Execution	152
12.	Functional Diagram of Security Macro Box	154

CONTENTS (Cont'd)

TABLES

<u>Table</u>		<u>Page</u>
1.	Assumed Equipment and Characteristics	27
2.	Hardware Added for Security	56
3.	ECP Service and Security Routines for User Program Control	82
4.	ECP Security Routines	99
5.	Tables Added to ECP for Security Purposes	100
6.	Hardware Techniques Considered for Security Protection	120
7.	Software Techniques Considered for Security Protection	124

GLOSSARY

ALPHANUMERIC

Character set including both letters and numerals and usually other characters. (American Standard Code for Information Interchange)

CONTROL CODE

A fixed length machine encoding of a control code name.

CONTROL CODE NAME

The English alphanumeric expression of security classification and any need-to-know restrictions for an entity of data or program.

CONTROL MODE

Mode in which a processor can execute the full set of operation codes.

DATA BASE

The store of information records being maintained for users; includes programs as well.

DESCRIPTOR

Instruction for input/output control processor execution.

ELECTRONIC DATA PROCESSING (EDP)

Data processing by equipment predominantly electronic.

ENTITY

A string of bits, characters, or words having an associated control code.

EXECUTIVE CONTROL PROGRAM (ECP)

Program that controls the secure execution of user programs by assigning hardware and performing security related operations

FAIL SAFE

Program or processing operation terminates automatically whenever proper responses to positive checks are not received.

FILE

A related information grouping, e. g. , logical records, card images, etc.

FLAG BIT

A bit contained in memory words and used for control purposes rather than actual user processing.

FORMATTED FILE SYSTEM

An information storage and retrieval system using a file design having fixed, periodic, and variable parts.

INPUT/OUTPUT CONTROL PROCESSOR (IOCP)

A limited purpose processor serving as intermediary between main memory and terminal units.

LOGICAL RECORD

A group of related items stored in one or more related physical records, depending upon length.

MODE

Processor condition as determined by state of a redundant set of flip-flops.

MULTIPROCESSING

Executing one or more programs simultaneously on more than one processor.

MULTIPROGRAMMING

Executing more than one program, time interleaved.

OBJECT

A contiguous string of instructions, data, or working storage required by a program.

ON-LINE

A terminal unit having direct connection with a unit buffer in the input/output control processor.

PERIPHERAL UNIT

Any type of input/output equipment connected with a unit buffer in the input/output control processor.

PHYSICAL RECORD

The smallest directly addressable portion of the data base.

PRIVILEGED INSTRUCTION

One executable by a processor only in control mode.

PROGRAM REFERENCE TABLE

Contains the name and/or descriptor for each object referenced by a program, and the base address and memory bounds for objects in high-speed memory.

SECURITY LEVEL

The maximum security classification authorized for information handled by an equipment, as determined by the equipment characteristics or its location.

TERMINAL UNIT

An input or output device in a work station.

THIN-THREAD ANALYSIS

Description of complex system operation or theory by following a single line, step-by-step, from start to finish, ignoring the secondary branches or ideas involved.

USER

Any authorized equipment operator, maintenance person, or intelligence research analyst. The system supervisor (or supervisors) is an authorizer as well as user.

USER'S CONTROL PROFILE

Completely describes each user's access authorization for information in the system in terms of control code lists by access type (read only or read and write). It also includes the user's key pattern information for identification plus authentication information for validating that the user really is who the user's key pattern indicates he is.

USER'S KEY

A physical card or key unique to a user which must be present in the user's key pattern generator at a work station to permit information flow with any terminal unit in that work station.

USER'S KEY PATTERN

An electrical logical bit pattern resulting from the user's key pattern generator at a work station which initiates user identification and is required for information interchange with any terminal unit in that work station for that user.

USER'S KEY PATTERN GENERATOR

A transducer from user's key to user's key pattern.

USER MODE

Mode in which a processor can execute only a partial set of operation codes; excluded are the privileged instructions.

WORK STATION

A separate, physically secure, area with its own user's key pattern generator in which the terminal units can be operated by only one user at a time.

SECTION I
SUMMARY ANALYSIS
AND
IMPLICATIONS FOR FUTURE PLANNING

BACKGROUND OF THE PROBLEM

The ever growing volume of classified military information required by the intelligence community is posing a serious problem for the near future. Efficient processing of large volumes of military intelligence will require a multiprogrammed, multiprocessing system with many input/output stations to be used by operators, analysts and programmers. Sharing such a facility among programs using data encompassing the full spectrum of classification levels while simultaneously servicing users of different communities of interest with limitations on access to one another's information presents many potential areas for compromising security.

Solution of the military intelligence data processing problem clearly rests upon establishing advanced security control techniques which form fundamental building blocks for development of the data processing systems necessary to meet future military intelligence requirements.

From an efficiency, flexibility, and cost-effectiveness point of view, it is desirable to have an intelligence data processing system which can easily and securely process any level of classified information without the need for specifying clearance equality between the physical facility and the data base information and with minimal concern for the security classification level of the operating personnel. That is, it is desirable to have a system which

permits personnel possessing varying levels of clearance to operate equipment handling information of various security levels without compromising that information, either by unauthorized access for which the user has no need to know or for which he does not have the necessary clearance.

Development of security control techniques fundamental to future modular electronic data processing systems is a basic purpose of this study. These techniques must be broadly applicable to intelligence data processing systems which are generally characterized as requiring a large data base having records with many different security and need-to-know partitions. The techniques considered are a combination of hardware additions and programmed controls. Each user of the system will have on-line access to the EDP system which provides controlled information storage and retrieval from the data base and processing as necessary to support his analysis or handling of raw data input.

In essence, Burroughs Corporation has undertaken this investigation to develop and evaluate advanced hardware and software techniques to ensure proper security control within a multiprogramming, multiprocessing system for electronic data processing of multilevel classified military intelligence.

STATEMENT OF OBJECTIVES

The prime study objective was to assure the safeguarding of multilevel classified (need-to-know) information by both software and hardware techniques in a single-level classified electronic data processing (EDP) system. Safeguarding included thwarting penetration attempts by unauthorized users to gain access to information, even in the event of hardware malfunctions. At the same time these access control innovations had to present minimal operational impediments to duly authorized system users. The effectiveness of this operational control was measured, therefore, in terms of the security protection provided as balanced against the restrictions on proper and timely information access.

The security techniques which were investigated, evaluated, and developed where promising were:

1. Centralized program control techniques which delegate the responsibility for security to the executive program.
2. Segregation techniques which physically isolate, in both storage and computational areas, data belonging in different security areas.
3. Redundancy techniques, both hardware and software, which independently verify security operations.
4. Authentication techniques (user's control profile authentication sequence only) which verify that the person requesting access has properly identified himself.

A second fundamental objective was to illustrate how the various evaluated and selected security processing techniques could provide building blocks for the development of an integrated intelligence data processing system that is workable within the constraints involved.

A final basic objective was to define future computer organization and/or programming in order to optimize the security efficiency-effectiveness relationships involved in a multiprogramming, multiprocessing computer complex.

Although these objectives were used to guide the study, it was clear that all phases of such a complex subject could not be definitively investigated within the time, money, and operational data restrictions involved. Therefore, certain constraints were placed on the study so that the critical areas could be explored in greater depth.

STUDY CONSTRAINTS

The security techniques study did not include cryptography or consideration of long-line communications problems. Also deliberately eliminated from the scope of the study were such areas as electromagnetic radiation, physical security, equipment wire tapping or physical modifications, personnel identification techniques beyond a user control profile system, and administrative procedures. Software design areas which were not to be studied included information storage and retrieval systems, operating systems, and file design (the formatted file system was to be assumed sufficient). On-line programming production techniques were to be de-emphasized, and it was established that the computer complex operating personnel should be considered as having security clearances equivalent to that of the data being processed.

Best estimates of the trends in the design of computer systems which are likely to have an important effect on intelligence data processing were used as a basis for achieving the stated objectives. Burroughs received requested inputs from RADC on this subject indicating that intelligence electronic data processing systems of the next generation will be modular and capable of multiprocessing and multiprogramming with time-shared access from many remote consoles. Clearly, it is impossible to attempt to specify completely the detailed parameters of future systems. However, a system design with the functional attributes just described was assumed, and many applicable techniques were developed and integrated into an overall system. Those aspects of a system design and configuration which would have significant effects on implementing the final recommended security control techniques are defined in the report along with those techniques which could be readily used with existing systems. Taking into consideration the objectives and the restrictions placed upon the program, five basic postulates were used to structure the study and methodology.

POSTULATES AND THEIR SECURITY IMPLICATIONS

1. The electronic data processing system has a multiprogramming and multiprocessing capability and includes remote on-line consoles.

The multiprogramming (more than one program time-interleaved) and multiprocessing (more than one processor operating simultaneously) configuration implies multiple units competing for memory access, multiple paths for information flow, and multiple programs or controller instructions concurrently in process. Such configurations have inter-unit conflicts for memory access with the attendant error possibilities of misidentifying the unit which has achieved access and misaddressing for access. Similar misconnections can occur in information exchanges among the processor units or with the associated peripheral units. Multiple programs present concurrently in high-speed memory must be provided with the capability for independence of data working storage and program segments as well as shared common program use on different data objects.

2. The system operates under an executive control program which performs security related operations in addition to its normal functions.

The executive control program (ECP) controls the operating system under which all user programs are run. The natural place for most software security checks is the ECP and the ECP-called service programs since they operate inter-mixed with user programs and control the execution of input/output operations and the critical registers and table changes for user programs.

3. Each system user has a uniquely defined security clearance and need-to-know classification limiting his access to the programs and data base.

The security clearance and need-to-know classifications authorized for a system user form the basis for the machine readable form of this information, hereafter called the user's control profile. Only the system supervisor(s) may have access to alter any user's control profile. No user (supervisor included) may alter his own control profile except under additional administrative controls.

4. A personnel identification technique exists which relates the current user of each equipment to the system.

The personnel identification technique is presumed to include for each user a visual recognition, a physical card or user's key for activating the user work station by making available a user's key pattern electrical signal for control use, and an authentication process which requires that a programmed private response sequence be entered by the user through the work station. The user's card or key is required to be present during work station use. Removing the user's key initiates a memory clearing process for the work station, its display images, and associated program data and working storage areas of the processors and memory. Work in-process for that user may be completed and its output shunted to a holding buffer to await re-identification of the user at the same or another work station.

5. Security techniques must impose only a nominal cost increment over the basic processing cost for new systems design; some techniques are applicable for retrofit.

The desirability of including any technique is based upon its effectiveness in providing security protection compared with its cost in hardware and processing time. The cost in a new system design should be reasonably small since the necessary features can be included from the start. The cost in most retrofit designs is magnified by the unavailability of conveniently located physical space, electrical capacity, spare logical gating inputs or fanout

capability, time slack in timing chains, excess memory bits, spare memory words, and by the requirement to redesign software.

APPLICATION OF SELECTED SECURITY PROCESSING TECHNIQUES TO A HYPOTHETICAL CASE

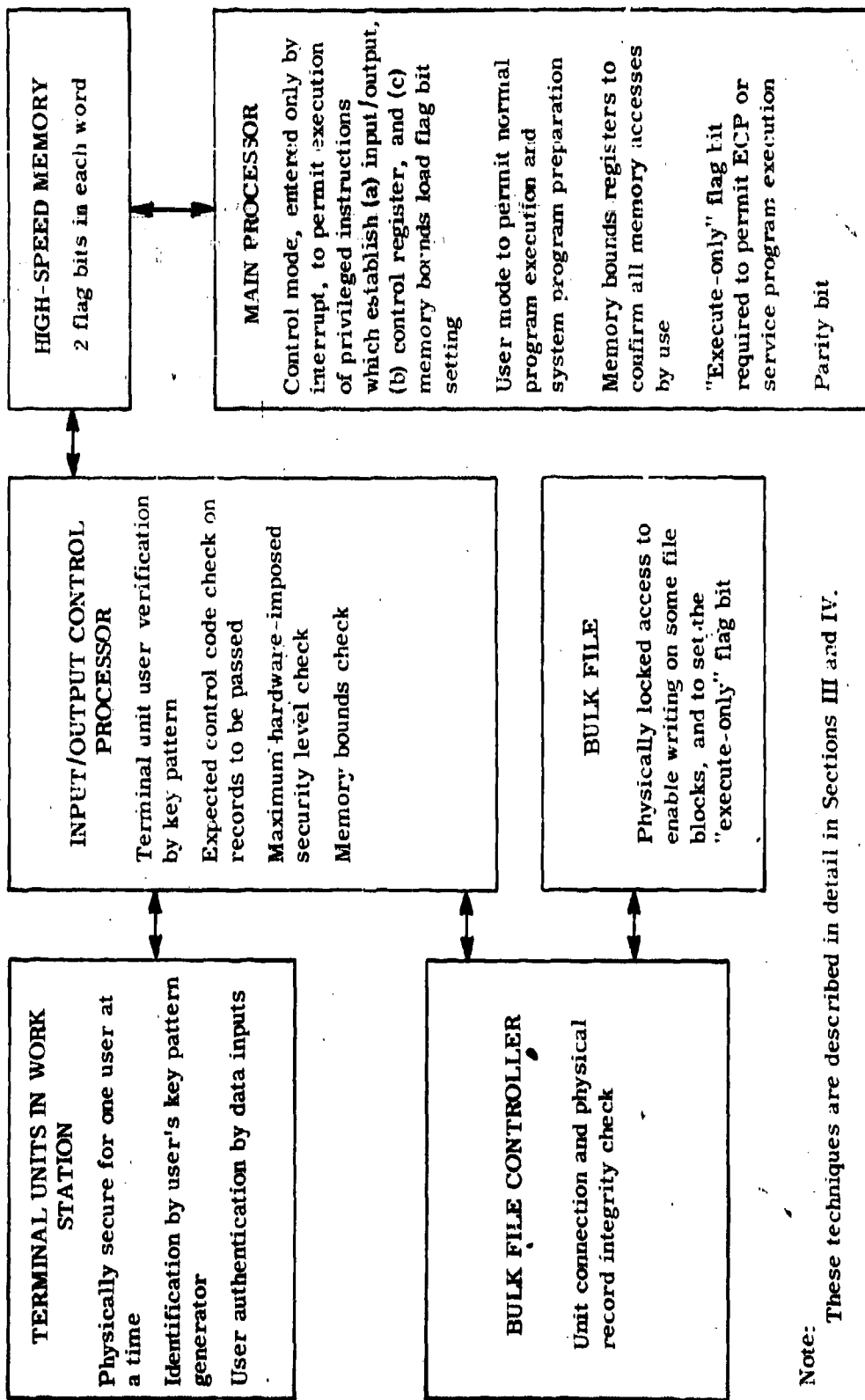
How the selected hardware organization and processing techniques are applied can best be illustrated by developing a hypothetical case as a descriptive illustration (Figure 1). This illustration demonstrates the workability of the selected techniques in a fail-safe computer processing system. A fail-safe system, in this hypothetical case, is defined as a system which permits an intelligence analyst to receive, process, store, retrieve, display, and internally transmit classified information to other users without compromising the information by either intentional or unintentional means. * For this hypothetical case analysis, all the facilities, equipment, and personnel involved are considered to have security clearance equivalent to the data being processed.

As shown in Figure 1, a set of basic operating procedures was established for processing data. These procedures are applied individually to the tasks of (1) input/output terminal processing (work station), (2) I/O control processing, (3) processing by the main processor, and (4) bulk file controller processing. Each of the above control procedures is implemented by a combination of hardware and software techniques. The recommended processing control combinations in Figure 1 are:

Input/Output Terminal Processing (Work Station)

1. A user's key pattern generator provides user identification.
2. Data input from the user provides further authentication.
3. The work station is physically secure and used by one user at a time.

* Implementation of the fail-safe concept is discussed in Section V.



Note:

These techniques are described in detail in Sections III and IV.

Figure 1. Interrelationships of the Recommended Security Control Techniques

Input/Output Control Processing

1. Terminal unit (console) user verification is made by continuing key pattern presence.
2. A control code check is performed on each record to be passed.
3. Each access to high-speed memory is checked against memory bounds to assure proper record exchange.
4. The maximum hardware-imposed security level is checked.

Processing by the Main Processor

1. Control mode, entered by interrupt only, permits execution of privileged instructions which establish input/output, control registers, and memory-bounds-load flag bit setting.
2. User mode permits normal program execution and system program preparation.
3. Memory bounds registers confirm all memory accesses by use.
4. Execute-only flag bit presence is required to allow execution of any word of ECP or service program.

Bulk File Controller Processing (including bulk file)

1. Integrity of each unit connection and physical record is checked.
2. Access to enable writing on some file blocks and to set the execute-only flag bits is kept under physical lock.

The recommended security control techniques are shown in Figure 1 as they occur in the representative hardware elements of the hypothetical modular, multiprogrammed, multiprocessor EDP system of the future. (For simplicity, only one element of a type is shown in this summary explanation. The security aspects of modularity are discussed in Section II.) The principal EDP hardware elements providing security protection are the terminal units in the work station, the input/output control processor (IOCP), the main processor, the bulk file controller and the bulk file which have just been described in terms

of possible processing control combinations. That part of the total cost of future systems attributable to high-speed memory is increasing, consequently, no memory changes are recommended for security protection other than two extra bits per word used as flag bits. The modularity trend for future processing systems permits the inclusion of parity checks on all information interchanges as a primary means for inter-unit hardware error detection.

The system considered has a continuing requirement that on-line users be granted access to classified information from the data base. The data base is stored in the bulk file and is accessed through the other hardware elements. In addition, the users may create information records for entry into the data base. This information storage and retrieval application includes processing of alphanumeric data as required to support the user's intelligence tasks.

Input/Output Terminal Processing (Work Station) for User Communication With the System

Security and need-to-know permission must be checked for all users who have access to information from the work station before automatic information release is permitted. However, if there is only a single work station serving multiple users having different security clearances and need-to-know authorizations, the system would restrict the amount of releasable information to the lowest security classification. This situation could well result in decisions being made which are unknowingly based on only a portion of the available information. It is recommended, therefore, that only one user at a time use a work station.* The single user sends his unique electrical user's key pattern signal to the input/output control processor (IOCP) and, upon verification of his identity to the EDP system after a suitable authentication procedure, is then eligible to receive all information authorized by his user's control profile (representing his security and need-to-know category).

* Modifications to this recommended single-user method to permit multiple concurrent users of a single work station are described in Appendix V.

Input/Output Control Processor (IOCP)

The IOCP is the connecting element between the many peripheral units and the high-speed memory. It executes descriptors, each of which provides the necessary control information for coordination of an input or output data transfer with an input or output peripheral unit. Descriptors are supplied to the IOCP from a specified region in the high-speed memory which is protected from user program access. Descriptors are prepared by the executive control program (ECP) which is not under user control.

The key pattern identifying the user of a terminal unit is passed through the IOCP to high-speed memory at work station opening. Thereafter the key pattern is used by the IOCP to assure continued user presence at the work station before any output is allowed. Each information record passing through the IOCP with the data base or to a terminal unit in a work station contains a header which includes the control code representing the security and need-to-know restrictions on the record. The IOCP matches this control code against the control code in the descriptor controlling the transfer. The classification described by the control code must not be higher than the hardware-imposed maximum level allowed for information to that peripheral unit. The set of maximum security levels is locked into the IOCP to prevent unauthorized alteration. The IOCP also checks that each high-speed memory address developed for accessing a new word of the I/O record is within memory bounds established to separate that I/O memory area from other concurrent users of high-speed memory.

Bulk File Controller and Bulk File

The bulk file controller receives control commands and serves as an address and buffering device for transfer of a physical record between the bulk file and the IOCP. It is recommended that a content check on each physical record

in the bulk file be made via the controller. A check word is developed and written with each physical record which is a function of both the actual file address and the content of the physical record. When the physical record is read, the check word assures that the complete contents have been accessed.

The data and program base are stored in the bulk file (typically disc, or tape). The system program (the ECP service and security routines) are stored in the bulk file and protected from alteration by switches which disable the write circuits to the file blocks in which they are stored. A second set of switches is provided to permit setting the execute-only flag bit of each word of these programs. Control of both of these features is a supervisory responsibility and is protected by a physical lock on a compartment of the bulk file which contains the switches.

Main Processor

The processor communicates with high-speed memory for program and data and executes the programs required by users. These user programs are controlled by the ECP and associated service and security routines. Two modes of operation are provided in hardware — user and control. The control mode is entered only by an interrupt. Control of interrupts is itself a control mode function. In control mode, execution of a few privileged instructions is permitted for establishment of input and output or to cause changes in control itself. This control includes establishing an interrupt mask register to selectively accept interrupts, to set the memory-bounds-load flag bit in a word containing the memory bounds for an area of memory, and to return to user mode. User mode permits normal programs to be executed, forbidding only the execution of the privileged instructions.

A program running in user mode may address only those memory areas allocated to it by the ECP. Memory bounds are established in hardware

registers about such a referenced area. Every developed address is confirmed as being within the memory bounds before memory access is granted.

For a system program to be executed in control mode, each word must have the execute-only flag bit set, as described in the section discussing the bulk file. The only way to set this flag bit is provided in a locked compartment of the bulk file. Thus, fail-safe confidence and control is achieved by the use of unalterable system programs (i. e. , unalterable by the user). These in turn provide security control over the user's programs run in the system.

The use of the techniques illustrated in Figure 1 provides the processing control procedures necessary for secure data processing operations. In turn, selection of the best combinations depends on the particular application. These techniques are described in detail and evaluated in Sections III and IV and their application is discussed in Sections V and VI. Illustrations are given on how these security checking procedures vary with changes in the relationships between user input and output terminal processing, input/output control processing , processing by the main processor, and bulk file controller processing. The results of studying these relationships substantiate two general conclusions.

1. Totally automatic processing control techniques that forestall all intentional or unintentional penetration attempts are desirable. However, such absolute control is not practical from a working point of view. The best resort is a secured system requiring that for a user to perform his intelligence analysis he must first satisfy several positive security checks; i. e. , several proper, logical, sequential commands and responses must be made and received before processing requests can be acted upon. Through the use of these self-checking hardware and software techniques, a fail-safe processing system can be created. These recommended

techniques can provide the necessary fail-safe assurances while at the same time providing sufficient operating flexibility to the user so that he can perform his job effectively and efficiently.

2. The security processing control problem can be further simplified by future multiprogramming and multiprocessing computer complexes in which some selected checking techniques that are now necessarily being performed by software are converted into hardware checks. Recommended examples include (a) memory bounds registers, (b) physically locked access to flag bit setting, and (c) bulk file content check. (Existing processors without control mode can be retrofitted with alarms associated with privileged instructions.) If these software checks are designed into the operational hardware, they will help achieve a computer complex specifically suited for the intelligence community.

These two general conclusions further support the study objectives by illustrating the overall effects of the security control processing interrelationships and how they provide building blocks for the development of an integrated intelligence data processing system. Also, if those previously described techniques are transformed into operational hardware, they will help optimize the secured processing efficiency - effectiveness relationships involved in a multiprocessing, multiprogramming computer complex.

Clearly, these conclusions indicate that there is still a fair distance between analytical modeling and operational workability. More study is needed to close this gap. Some suggestions on "how" are presented next.

IMPLICATIONS FOR FUTURE PLANNING.

The completed study clearly substantiates that secured processing efficiency will be improved over existing methods by implementing the recommended

security control checking techniques. The degree of improvement is a function of the information storage and retrieval concepts and operational procedures involved in any given application. Security aspects of the data storage and retrieval procedures are significant both to the protective effectiveness and to the processing efficiency. For example, the cost effectiveness for a control code attachment as a header for each entity* is determined by many specific factors that make up the selected operational data store structure. These factors are:

1. the number of different control code names required,
2. entity sizes,
3. statistics, by control code names, showing the frequency of occurrence in the data base, and
4. frequency of access to these entities.

Detailed evaluation of these factors through empirical data collection and analysis is required and necessary to measure the amount of error protection and efficiency achieved within the information storage and retrieval system under investigation.

We have assumed during this study that the input processing consists of raw data as well as evaluated information for further processing and filing. It follows, therefore, that a data entity is likely to be large by comparison with the control code representing its security classification. Since the quantities of data to be handled will eventually reach huge proportions, information retrieval is an increasingly complex problem. It is probable that an intermediate file would be necessary to index the main file, perhaps using abstracts or key-words. Data entities might thus be so limited in length that the security control

* Entity is defined as a string of bits, characters, or words having an associated control code.

code would actually become the larger element. In this case, a more economical means for appending security information to the data would be required. Probable solutions are the use of variable length codes, optimized as to frequency of data use, or format representations of classification and need-to-know. Determining the particular values of these factors is a subject that needs additional study.

The increased data processing capability of future EDP systems permits consideration of remotely located on-line work stations. Remote work stations imply requirements for communications security. At the EDP center, a common on-line cryptographic processor would provide an economic advantage compared with separate cryptographic equipments for each line to a work station. Given the common cryptographic processor for communications security, it would also be used as a super-encryption device for particularly sensitive information stored in the data base. Further study of shared use of cryptographic processing equipments is required to identify economies in their use as a security protection adjunct to an EDP system such as is recommended here.

These potential development areas include identification and avoidance of compromises in memory hierarchies, personnel authentication techniques, and techniques for processing encrypted information. The latter area lends itself better to multiprocessor systems than to the present sequential single-processor systems since an individual job may be split among many processors so that its control is not known to the user.

SECTION II

PHYSICAL SYSTEM CONFIGURATIONS

The preceding discussion attempted to provide a summary of study achievements and the recommended combination of hardware and software security techniques that will produce a fail-safe system. Substantive material to support these recommendations is described in detail in this and following sections. This section contains a sample physical description of a modular, multiprogramming, multiprocessing system which represents the next generation of an EDP complex. The recommended physical configuration is described, along with an indication of the hardware features and their use by software to provide security control. The recommended hardware features are detailed in Section III and the software features in Sections IV and V.

The apportionment of security protection techniques between hardware and software is implied by any description of a physical configuration for the system. In general, those aspects of security protection which require highly repetitive operations are candidates for hardware and those which require decisions based upon many changing conditions are candidates for software.

PROCESSING SYSTEMS ANALYZED

To lay the groundwork for a meaningful later consideration of the specifics of security protection, it is necessary to define the types of equipment which will be under discussion. To this end, electronic data processing (EDP) equipment

has been divided into three broad categories: present-day equipment, evolutionary equipment which can be foreseen in the near future, and developmental equipment of the more distant future based upon an assessment of developmental trends as they appear to Burroughs.

Present Day Configuration

A present-day EDP configuration for intelligence application can be characterized as follows. The hardware includes a central processor, high-speed memory, input/output channels for linking with conventional peripheral card units, magnetic or paper tape units, high-speed printers, and controllers for linking with random-access bulk files in which the on-line data base resides. This equipment configuration provides EDP in a batch processing mode with a programmer intermediary between the analyst and the hardware. Job turn-around time is generally measured in hours. Because of the long turn-around time, each job processed at user request tends to be large, with bias toward retrieval of too much information processed in too many ways. This often results in extraneous hard copy output from which the analyst selects the pertinent part. Security protection principally resides in physical control and personnel security clearance.

Evolutionary Configuration

A significant step in EDP system evolution is to provide analysts with on-line consoles and time-shared (not high-speed memory space-shared) use of the data processing equipment. The purpose is to reduce turn-around time between desired action formulation by the analyst and access to the pertinent information available in the system which is processed as he desires, and is subject only to the recommended security controls. This system differs from

the recommended system principally in the use of only the single time dimension of shared facility use. Significant overhead processing results between user jobs. In order to minimize this overhead, each job should run to completion (i. e., the present batch processing). This is incompatible with the objective of many on-line users, consequently, the compromise in time sharing takes the form of frequent temporary terminations of partially completed jobs so that each user appears to get full access to a less powerful machine (i. e., project MAC). The interchange cycle is limited by memory swap time considerations to about 10^6 memory access times so that a significant amount of processing is done between swap times (at least 50 percent of total processing should be production). With as few as 16 users, this amounts to a several second cycle to service all users for present systems.

The evident conclusion is that memory swap-time and inter-user overhead processing should be reduced to achieve more productive processing. Both are achieved in the recommended system by allowing high-speed memory space sharing as well as time sharing.

Modular, Multiprogramming, Multiprocessing Configuration

The system configuration representative of future EDP systems for the intelligence community is shown in Figure 2. The basic processing elements shown in this diagram are the terminal units contained in physically secure work stations, the input/output control processors (IOCP), the high-speed memory modules, the processor modules, and the bulk file controllers and their associated bulk files. Special purpose processing equipment is minimized for data interfaces handling common characteristics. Sufficient quantities of each element are provided to fulfill the system demands; the indicated

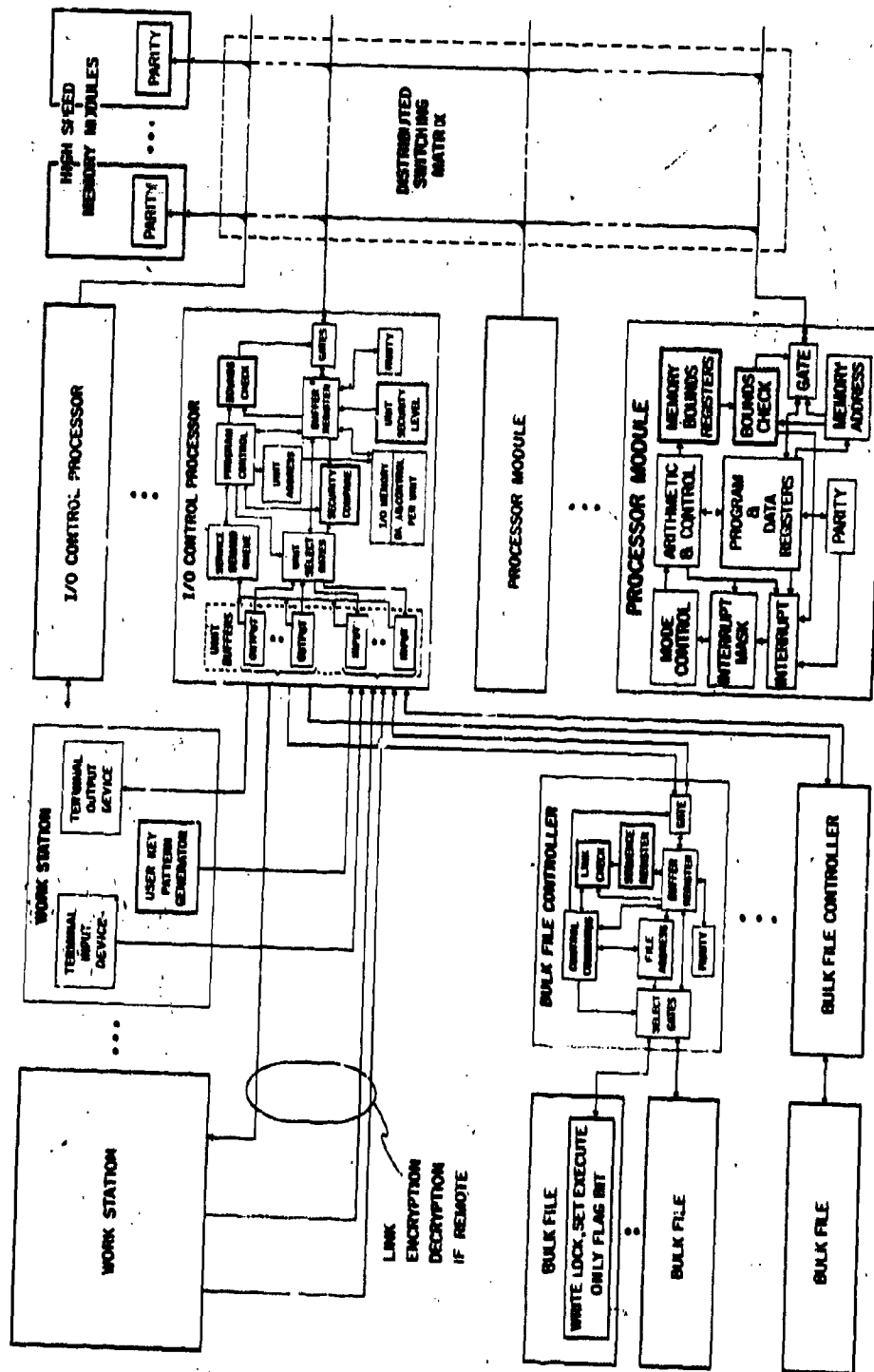


Figure 2. Block Diagram for Future Multiprogramming, Multiprocessing EDP System

complement (two elements of each type) is sufficient to demonstrate the security aspects of multiprogramming and multiprocessing in a modular system.

The work stations provide private working areas for individual users. They include terminal units necessary for user input, off-line processing (such as editing), and user output. The private work station permits release of any information to that work station, limited only by the current software user's control profile opened for the user of that work station and by the physical maximum security level of the work station or its included terminal units.

The control programs executed on the processors provide the parts of security control requiring flexibility at least cost. A number of processor design features to simplify this control are recommended. Memory bounds registers constrain user program addressing to those particular areas of high-speed memory previously allocated to it. A control mode, entered by interrupts, is recommended in which privileged instructions are executable (to allow changes of control and establishment of input or output). The ECP and associated service and security routines run in control mode. The normal mode is the user mode, in which almost all processing is performed. Control sensing of the two recommended flag bits permits identifying memory words whose contents may be used to alter memory bounds registers and to recognize control programs.

In an information storage and retrieval application when most jobs are limited by frequent data base accesses, several jobs can be concurrently processed. To overcome overhead burdening of the evolutionary time-sharing system,

sufficient high-speed memory can be provided to permit several partially processed jobs to remain dormant yet available for re-initiation with minimum overhead. The several jobs tend to minimize the unmasked input/output, particularly if the active job mix includes some job which is processor bound. Thus, the recommended system includes both time- and space-sharing of high-speed memory.

The shared use of high-speed memory on a dynamic basis requires positive assurance of restricted access to only those areas allowed for a particular user program. Hardware memory bounds provide independent confirmation that any address developed by a user program falls within an area allocated to that user. Such bounds are recommended in both the processors and IOCP. Restricting control of these memory bound registers from user access effectively blocks the user from other areas of memory for which he has no need. Thus, he has no access to other users' programs and data and no access to the ECP, service, and security routines.

Dynamic memory allocation requires symbolic addressing to named objects so that no user need know any absolute address. The ECP or service program controls allocation. Similarly the information storage and retrieval system controls all accesses to named logical records in the data base. No user need know any actual location of a logical (or physical) record in the bulk file.

The other basic element for security control in hardware is the IOCP. It is the intermediary element between the high-speed memory and output to the user in a work station. It is also the intermediary between the high-speed memory and the bulk file in which the data base is stored. In this intermediary role, connection checks are made for each information exchange to assure that

the proper peripheral units and high-speed memory blocks are linked. In addition, security comparisons are made against (1) the maximum security level allowed for the particular peripheral unit, and (2) the actual control code expected for each record being passed.

The bulk file controller serves an intermediary role between the actual mass storage on the bulk file and the IOCP. It accesses one physical record at a time from the bulk file. In order to ensure access to an entire physical record it is recommended that a link check word be added to the physical record when written and be checked when read. The bulk file includes a physically locked compartment in which switches are provided so that at least some blocks may have the write circuits disabled, and may have an execute-only flag bit set. Into these blocks would be stored the ECP, service and security routines to protect them from alteration and to permit a supervisor to authorize the ECP, and routines for execution to control the system and provide the intended security safeguarding.

In the recommended modular configuration, as in the evolutionary configuration, information available for control of the data base resides in controlled limited access to file directories, and in control codes associated with data entities. The control information includes a user's control profile which has been currently established as active for a work station and its included terminal units by both the user's identification-authentication process and the internally associated user's control profile identifier appended to the job. An equipment maximum security level table is included if there are equipment-imposed security restrictions. Finding the data control code in the user's control profile and using equipment of adequate security level are prerequisites for preparing data for user output. In addition, user presence in a work station with user's control active is a prerequisite for actual output release to that equipment or any terminal unit in that work station.

INFORMATION FLOW EXAMPLE

The information flow for a typical work action by a user in this system includes the following steps relative to Figure 2.

- 1. The user identifies himself in a work station, resulting in a private user key pattern electrical signal representing him to the IOCP.**
- 2. The IOCP interrupts some processor and provides the key pattern and work station identification.**
- 3. The ECP running on that processor interprets the interrupt and responds with an authentication challenge back through the IOCP to the user which is displayed on some output terminal unit in the work station.**
- 4. The user enters necessary authentication data sufficient to allow the system to open that user's control profile and thereafter permit information processing on records for which the user's control profile indicates security clearance and need-to-know.**
- 5. The user enters processing requests using a query language (procedure oriented) or symbolic programming language.**
- 6. These processing requests are converted into user programs under control of system software (the ECP and appropriate service programs, compilers, assemblers, etc.).**
- 7. Execution of a user program generally requires access to the data or program base stored in the bulk file. The information storage and retrieval (ISR) software system provides this access for the**

user program. Each data base input and output through the IOCP with the bulk file is accomplished for the ISR program by means of a service program not subject to user programming alteration. The ISR program checks the control codes of each retrieved record against the requesting user's control profile as a prerequisite for release of the record to an area available to the user's program.

8. The result of processing by a user program is generally some output to that user. This output is handled for the user program by a service program not under user control. It includes security checks against the control code of the information.
9. The IOCP checks that the same user key pattern is still present in the work station as is indicated in the header of the output. It also confirms the connections to both terminal unit and high-speed memory address. Successive physical records of the output are similarly checked to ensure integrity.

SECURITY PROBLEM AREAS

The additional switching points required to allow module interchangeability in modular EDP systems pose a potentially greater security problem. In Figure 2 the principal switching points are the distributed switching matrix connecting the high-speed memory modules with both the IOCP modules and the processor modules, the unit select gates in the IOCP, the bulk file select gates in the bulk file controller, and the addressing networks in each high-speed memory module and bulk file. A hardware error in any of these can result in misconnection and thus potential security violation. Two principal solutions to this risk are: redundancy - including parity checks on all information (address and data) transfers, and including sufficient differences in codes representing

addresses that any single bit error will be detected and will result in a fail-safe error investigation; and feedback - providing an identified connection confirmation signal which must match that signal expected for the connections prior to actual information release.

There is little justification for cryptographic techniques for communications security in the system, since the inter-equipment cabling is assumed to be contained within the physically secure facility. However, as the processing capability expands to allow effective handling of larger data bases, need for (and economic justification of) remote on-line work stations should increase. At such time, on-line link encryption-decryption may be required. The cost of cryptographic units, supplied one pair per link, grows with the number of links so protected. Beyond a set number of links, provision of a central cryptographic processor shared among the links becomes economically attractive to replace the separate per link units at the EDP complex. Added protection against missending can be achieved in this case since the receiving link decryption device could not produce clear text from a record encrypted for a different link and the transmission output would be garbled. Consideration of cryptographic techniques for other applications within the physically secure facility, given a shared cryptographic processing capability, was considered beyond the scope of this study.

Table I summarizes the hardware units that are used to develop a modular, multiprogramming, multiprocessing system configuration. A modular system made up from multiple interchangeable units of each type is permitted. Processing features for each hardware unit are identified and those features added primarily for security protection purposes are noted. Additional features which provide an increasing amount of security protection are designed into the multiprogramming, multiprocessing system. All these hardware aspects are discussed in Section III.

Given a physical system configuration containing the equipment listed in Table 1 as a frame of reference we can meaningfully investigate those hardware security techniques which aid in the development of a secured system. Detailed descriptions of those techniques and how they interact are presented next.

Table 1. Assumed Equipment and Characteristics

<u>Unit And Features</u>	<u>Extra For Security</u>
<u>Processor Module(s)</u>	
Single arithmetic and control unit	
Word and character operations	
Two modes: user and control	
Interrupt control through mask	Interrupt response in control mode
Flag bits	Control programs different than user programs
Memory bounds registers by function: read only, read and write, and execute	Usage control of read and execute, register loading by flag bit protection
Memory addressing: base relative, indirect, may be indexed	Intermediate addresses bounds checked
Privileged instruction macros	Predecessor-successor link checks, control mode
<u>Memory Module(s)</u>	
Independent address control in each module	
Access priority resolution among IOCP's and processors	
Parity checks of input addresses and data, parity checks and passing on output data	

Table 1. Assumed Equipment and Characteristics (Cont'd)

<u>Unit And Features</u>	<u>Extra For Security</u>
<u>Input/Output Control Processor(s)</u>	
Buffers and services interrupts from peripherals for processor	
Data buffer between peripherals and memory	
Memory address control including bounds check	
Peripheral connection control by selected unit buffer	
Parity checks	
Security verification	Expected content comparisons against control code or user's key pattern
Unit security level set and compare	Maximum security level of terminal unit
Memory erase compare	Block erase and verification
<u>Terminal Unit(s)</u>	
Data Exchange with IOCP unit buffer	
Unit identification	
Interrupt generation	
Parity generation, check	
Link encryption-decryption	As required for remote unit
<u>User Work Station(s)</u>	
Group of terminal units	Physically secure enclosure used by only one user at a time
User's key pattern generator	Terminal unit for user's identification and work station use verification.

Table 1. Assumed Equipment and Characteristics (Cont'd)

<u>Unit And Features</u>	<u>Extra For Security</u>
<u>Bulk File Controller(s)</u>	
File address control	
Parity checking	
Unit connection identification	Parity protected for error detection
<u>Bulk File(s)</u>	
Physically locked enclosure	Opened only by supervisor
Write lock	Segments protected from overwriting
Execute-only flag bit set	ECP identifier

SECTION III

HARDWARE SECURITY TECHNIQUES

Data processing system security violations, whether caused by hardware or software failure, occur in either of two technical problem areas:

1. Compromising malfunctions
2. Unauthorized access

A "compromising malfunction" results from a system malfunction which causes the presentation of data to someone who is not authorized to receive it. This type of violation would occur if data intended for transmission to a (then) secret station were misrouted and sent to a (then) unclassified station. By "unauthorized access" is meant the type of security violation which occurs when someone is able to "outsmart" the system, and, without authorization, enter into or read from the system information which is classified, unclassified, simply accidentally, or whatever. It is just as serious to allow unauthorized personnel to enter intelligible data, whether classified or not, and, hence, affect decisions made by authorized personnel, as it is to allow them to receive data from the system. It is helpful to make a distinction between these two types of security violations since security control checking procedures which reduce the probability of one type of violation do not necessarily reduce that of the other.

The basic problem to be attacked (Sections III and IV) is that of assuring security in a data processing system which is simultaneously processing and simultaneously storing data of many different security classifications.

As stated in Section I, the security classification of the computer complex itself remains fixed and is the highest classification ever associated with the data to be processed. Around this centralized computer complex are various peripheral input/output work stations which have varying security classifications. Security classification of any one station can change with time and may vary over the entire range of possible security classifications. Although not considered in this study, these peripheral stations may actually be thousands of miles from the computer complex. Conventional link encryption and decryption techniques are used for any remotely located devices. Stations can vary all the way from a single flexowriter to an entire computation center. Thus, the study considers these peripheral stations in as generalized a manner as possible, and the conclusions obtained are as independent as possible of the remote station characteristics. The centralized computer complex may itself be divided into smaller, separate units, but the logical, functional result can be considered as a single computer complex in one location. Nevertheless, in this study, work stations are considered to be located adjacent to the computer area.

The hardware security techniques recommended for inclusion in a modular, multiprocessing, multiprogramming system are described on the following pages. Descriptions are organized about the system equipment and security features described in Table I, Section II. Section VIII summarizes in table form all hardware security techniques considered during the study, whether recommended or discarded.

PROCESSOR

A processor module has basic arithmetic and control capability and operates on words or characters. Parity generation or checking is provided for all memory accesses. Most of the features used for security protection are identified in Figure 3. The processor has two modes of operation, control

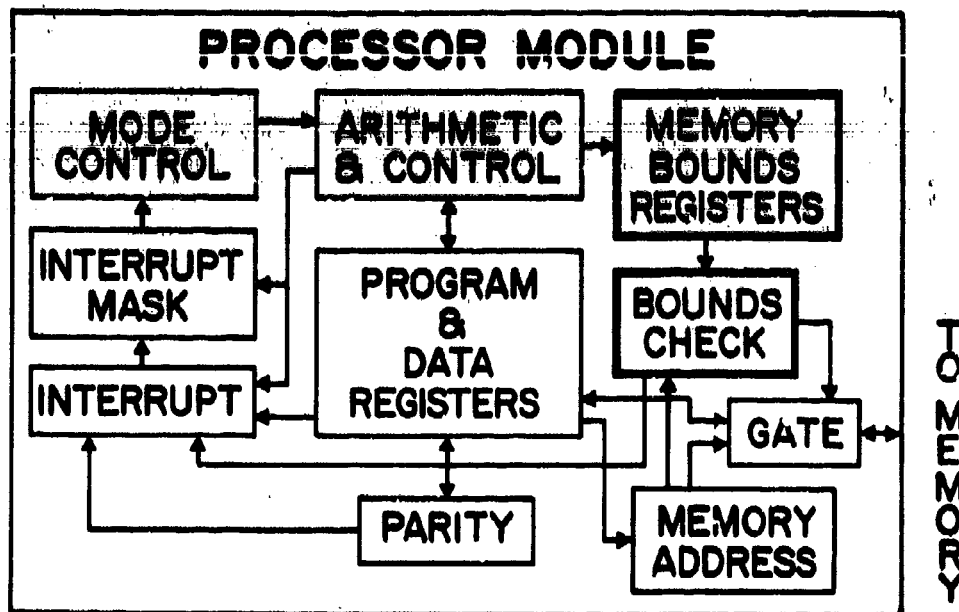


Figure 3. Processor Module

mode and user mode. The control mode is entered through a maskable interrupt system, and flag bits are included in the processor for control of word content while in user mode. The processor module contains privileged instructions usable only in control mode, and includes memory bounds registers providing comparison for every generated memory address to restrict user program access to those memory addresses previously verified as necessary for that user program.

Processor Modes

Processors now in development (IBM System 380, GE 600 Series, and Burroughs B8500) each have multiple modes of operation differing in the ability to process available instructions and in memory access restrictions. For intelligence

systems, the modes should be separated into at least two groups: control and user, differentiated by the setting of a redundant set of flip-flops.*

In the control mode, the full instruction set is available. This includes both unrestricted instructions and privileged instructions. Examples of privileged instructions include those which control input and output, memory bounds registers, and interrupt mask register loading. Privileged instructions are described on page 38. Most instructions are unrestricted.

User programs are executed in the user mode wherein only the unrestricted subset of the available instructions can be executed. Any requirement for a function performed by a privileged instruction is fulfilled by a call on the executive control program (ECP). Should a privileged instruction occur in the user program, it is treated as illegal and results in an interrupt transferring control to the ECP. The processor responding to this interrupt is placed in the control mode; the interrupt is interpreted by the ECP and results in execution of the necessary control action (described below). Proper completion of this control action is a prerequisite to executing the privileged instruction desired by the user program. Upon completion of the privileged instruction, and any subsequent ECP tasks, control is restored to some user's job (possibly different from the one which caused the interrupt) and the processor module is returned to the user mode.

The control mode may be further partitioned to provide more than one level of control in which different restrictions are placed on using privileged instructions, addressing reserved memory areas, and responding to interrupts.

*The mode switching circuitry should be sufficiently redundant to negate the probability of hardware failure causing mode control error. Appendix IV is an annotated review of work devoted to achieving reliable machine operation. Inclusion of a fail-safe control mode signal is a logical requirement for executing control-mode-only functions such as privileged instructions.

For example, service programs would generally be loaded as needed by the ECP into an area of memory. A mode is possible in which input/output setup instructions could be executed from this area of memory without full processor interrupt for ECP control.

User-programmed entry into control mode is only by an unmasked interrupt as described below. Programmed exit from control mode back to user mode is accomplished by executing a return-to-user-mode (privileged) instruction. Any programmed attempt to circumvent the interrupt entry is prevented in normal operation by physical lockout of access paths and flip-flops and by execute-only flag bit identification of the ECP and service programs, as described on page 36. These precautions provide assurance that the control of interrupts resides in the ECP.

Interrupts

Control mode is entered only by means of an interrupt generated by some processor. An interrupt results from one of many possible conditions arising either internally or externally to the processor. An interrupt sets a specific condition bit in an interrupt register. Interrupts may be masked in order to control the priority of servicing, to select which processor performs the servicing, and to allow completion of processing one interrupt without being further interrupted. An unmasked interrupt condition results in entry to control mode and control program initiation at a specific location unique to the interrupt. A masked interrupt is held in the interrupt register until the masking bit is cleared. Typical interrupt conditions include external input/output activity requests, abnormal arithmetic or logical conditions, memory bounds violation, power failure, equipment errors, occurrence of a privileged instruction in programs while in user mode, real-time clock updating, and interprocessor communication.

Preferred identification of an interrupt involves assignment of individual bits to specific causes. Since no alternative way of setting the bit is provided, a unique meaning can be associated with it. If coding is required to reduce the number of bits representing possible interrupt conditions, redundancy (such as parity) should be employed to reduce the probability that a hardware malfunction will cause faulty decoding.

Flag Bits

Flag bits are contained in memory words which are used for control purposes rather than for actual user processing. As such, the bits are not alterable by the user but are inserted by special physically unlocked switches, by the ECP, or are created by hardware alone. Many applications of flag bits are possible. The most significant flag bits for security protection are described in the following paragraphs, with the first three, parity, execute-only, and memory-bounds-load, being recommended.

Parity is assumed as a minimum flag bit. Parity is generated by all units preparing information for transmission and is checked by all receiving units. This applies to both control or address transfers and to data transfers. A single parity bit detects any single (or odd number of) bit error(s) in the word (character or bit group) in which it is included. Multiple parity bits per word are not generally required.

An execute-only flag bit is used to identify the ECP and service programs. For a processor to execute any word of such programs, this bit must be set. Bit setting is protected by the same physical lock in the storage module of the bulk file that protects the "write" lock through which programs were originally written into the bulk file. This is more fully discussed on page 54.

A memory-bounds-load flag bit indicates a memory word from which a user's program can access an object and have the memory bounds registers loaded to bound the object. This flag bit may be set only in words of a program reference table (PRT). It is set using a privileged instruction by the ECP after it has allocated a memory area to the object required in the PRT and after it has set and checked the protection provided by the memory bounds. When the ECP is ready to release a processor to execute a user program, it loads the PRT bounds registers with the PRT bounds for that user program then returns that processor to user mode under control of the bounded PRT. The PRT contains both the memory base addresses and memory bounds for all objects which the user program requires. Each time the user program references a different object from its PRT, the memory bounds registers will be set about that object so long as the memory-bounds-load flag bit is set in the PRT entry for that object. Successive accesses to the same object do not require resetting the memory bounds register.

Since the memory-bounds-load flag bit can only be recognized when it is part of a PRT word (accessed between the PRT bounds), its use as a unique flag bit is redundant. If total hardware trust is placed in the PRT bounds register and parity checks are made on addresses, the bounds-load flag bit could be replaced by a specific normal bit of the PRT entry. The choice between these two approaches depends upon specific hardware design. The flag bit is recommended, even though the extra redundancy is significant (one extra bit of each memory word), since this is a crucial item for which undetected failure leads to serious security violation. Another alternative to the memory-bounds-load flag bit is to make memory bounds loading a privileged instruction executable in control mode only. This is not recommended because much more ECP overhead time is required.

A jump-trap flag bit can be used to designate the end of a data block or a dynamically sized stack extent limit. Its effect is to force an interrupt when the bit is recognized. Application of this bit is limited to those cases where each successive word is addressed without the capability to skip addresses. The jump-trap flag bit is not of prime security importance, although it allows many iterative control programs to automatically terminate or allows recognition of the flag bit without an explicit condition being tested in each loop of an iteration.

A security-label flag bit can be used as a classified-unclassified indicator. For some applications requiring only a few control codes, say X , a small number of flag bits could be assigned as a per-word classification indicator. If $2^{n-1} < X \leq 2^n$, n bits would be required to fully express the control code in the security label. For the system considered, X is of the order 1000, so the number of bits for this purpose is unreasonably large: 10 compared to the 48 to 64 information bits expected per word.

Privileged Instructions

Privileged instructions are used to establish or alter the overall control by the processor of user jobs. Therefore, for a system programmer to separate the security controls illegally he must use privileged instructions. Security control is thus dependent upon control of privileged instruction execution. The best way to provide this control is to require the processor to be in control mode as a condition for execution of privileged instructions. Logical implementation requires the presence of both the control mode signal and the decoded privileged instruction signal before allowing execution of privileged instruction.

Privileged instructions include:

1. Input/output command descriptor establishment for later use by the input/output control processor to control information transfer,
2. flag bit setting on memory bounds loading information,
3. PRT bounds register loading,
4. interrupt mask register control,
5. interrupt response base address loading, and
6. mode control register resetting to return to user mode.

A privileged instruction occurring in a user program is treated as illegal and results in an interrupt without execution. This interrupt, like any other, results in entry to the control mode to extract its appropriate response (in this case analysis, logging, and recovery or termination of the user program).

Fail-safe recognition of privileged instructions can be aided by proper choice of operation codes. The relatively few privileged instructions could all occur in one group, members of which must have at least two bits different from any members of the group of non-privileged instructions. For example, with a 6-bit operation code there are 7 combinations having 5 or 6 ones. These should be ample for all privileged instructions. There are 15 with 4 ones and 42 with less than 4 ones. By using those having 4 ones for error detection, the desired object is achieved while only sacrificing 15 of the 64 possible operation codes for error detection. To achieve comparable protection by assigning a single flag bit to denote privileged instruction would, in this example, require a seventh bit for operation code if more than 25 non-privileged operation codes were required. For 8-bit operation codes, the comparable numbers are 9 for privileged, 28 for error detection, and 91 non-privileged.

In Appendix II, external implementation of an alarm associated with privileged instructions is described. It is suitable for retrofit to a processing system which has no control mode.

Memory Bounds Registers

The combination of memory bounds registers with a fail-safe control program is the recommended means of providing memory access control. Other considerations and methods for memory access protection are given in Appendix III. In the assumed multiprogramming, multiprocessing environment, floating programs are necessary to permit dynamic memory allocation for space-sharing of memory among multiple concurrent users. Memory access control is further complicated by multiple ways of addressing.

Assume that processors include the common techniques of floating programs:

1. Relative addressing with respect to:
 - a. Base (or index) registers set by the control program
 - b. Base (or index) registers read by the user program from a program reference table
2. Indirect addressing via a program reference table.

The principal feature of the hardware recommended for checking addresses is a group of memory bounds registers. For this discussion a memory bounds register pair defines both upper and lower bounds, and also specifies allowed use: execute-only for a program area, or whether read-only or both read and write are permitted for a data area.

Each address developed in a user program (whether for program, indirect address, or effective address of an instruction or a data object) is compared against the appropriate memory bounds register pair to assure that the address is within the bounds

To assure address control of floating programs, the system processor hardware must provide at least one control mode in which the ECP performs memory allocation to match user programs to the space available. Allocation includes assigning actual memory addresses and bounds on the areas of named objects required by a user program. The program reference table (PRT) contains this information, one entry per object. ECP software checks the allocation and effectiveness of memory bounds before releasing a PRT to a user's program. A privileged instruction sets the memory bounds load flag bit in a PRT entry. This flag bit allows the PRT entry to be used to alter the contents of the second type of memory bounds register described below.

The first type of memory bounds register bounds the PRT. Two are employed and allowed a user's program to reference by indirect address the arbitrarily allocated objects required by the program. Such a reference permits access by the user's program to the base address of the named object area and, thus, internal entry access by indexing relative to that base. In order to provide memory bounds about the memory area allocated to the referenced object, the reference must be to its PRT entry by indirect address. Indirect addressing is indicated by an indirect address bit in the program instruction. The memory bounds flag bit in the PRT entry must also be set. Only one pair of PRT bounds registers is required by a processor since only one program is executed at any one instant, and that program generally runs for a significant period (milliseconds) so that the overhead for reloading this register pair is negligible.

The second type of memory bounds register is used to indicate the access permissible in a particular memory area. The types considered are (1) execute-only -- for program strings, * (2) read only -- for data tables which are to be

* Multiple concurrent users of a single copy of a shared program allowed in multiprocessing systems require that the program remain unmodified while being executed; address alteration by indexing, or indirect addressing through a data word, are permitted since neither modify the actual program.

referenced but not altered, and (3) read and write -- for working data storage. Two bounds register pairs of the second type are recommended, one for the program string in process and one assignable to either of the data access types. Determination of which is to be loaded is a function of the hardware and the PRT entry resulting from the instruction making the indirect address call.

Any address developed in the user's program which is outside the three areas for which bounds are provided or any access attempt within the bounds areas of the wrong type results in an illegal access interrupt before completing the memory access. No way exists for a user program to alter the memory bounds established for it by the ECP. Thus, in the absence of hardware failure, the user program is confined to the areas which the ECP grants access. Hardware failure is principally guarded against by parity checks and frequent software confidence checks. An example of such checks is the one recommended for each memory allocation to ensure that the memory bounds actually work as intended.

It is not recommended that memory bounds independent of the normal memory bounds applied to user programs be applied to provide redundant protection to the ECP and its associated tables. A processor can only be in one mode at a time, therefore, in control mode, the normal bounds are superfluous since the ECP must have general access to memory in order to perform its required control functions. In user mode, the normal bounds are established to surround and define access blocks. Addressing outside these blocks causes alarm interrupt.

Control bounds on the ECP provide a second check should hardware or software malfunctions cause missetting or misaddressing into the ECP area. Since there are greater security risks inherent in addressing into another user's information

concurrently in memory, there seems to be little advantage in specially protecting the ECP and its tables by special memory bounds. Instead, the ECP and service programs are identified by flag bits. Memory bounds are also required for the input/output control processor where they function to allow either reading or writing access within an area of memory.

A single memory bound register can provide one boundary across storage. By interpretation, this can represent a lower bound for one program and an upper bound for another program (with the actual address bound assigned to one or the other, but not both). This bound can have precision of the significance assigned the least significant bit and range equivalent to that describable by the number of bits in the register. If the precision is blocks of 2^n words (characters where $n = 1, 2, 3, \dots$), then storage allocation and memory bounds protection is by an integral number of entire blocks.

If the register length is less than that required to address the entire memory at the given precision, the highest order bits must be furnished from some other addressing register which essentially provides the storage module selection. These high order bits provide fixed memory partitioning into modules and can be used to provide fixed-location memory bounds. The number of registers required is dependent on the storage size, the limitations on assignable space, and the block size.

The cost of one bit of a hardware memory bounds register, operating with negligible time added to a memory cycle, is approximately \$50 (1965). This cost assumes five integrated circuits (a flip-flop, a comparison circuit, and associated gating).

Within five years, this cost should reduce to about \$5 for a single integrated circuit which achieves the above functions and contains one bit. The cost grows

nearly linearly with the number of bits, and for meaningful register sizes, dominates the cost of the error alarm circuit following the comparators (no more than four integrated circuits).

Processor Hardware Access Restriction

Protection against unauthorized access to the special hardware used only in control mode can be achieved by either isolating it into locked compartments, by sealing it in place once installed, or by providing an alarmed interlock. Maintenance in any case would require supervisory concurrence.

The processor hardware used only in control mode is a relatively small part of the total processor hardware, and includes:

- a. mode flip-flops,
- b. privileged instruction decoding logic,
- c. ECP base program register,
- d. PRT memory bounds setting logic,
- e. interrupt register and its associated mask,
- f. I/O-descriptor-enabling logic, and
- g. flag-bit-setting logic.

Providing physical isolation of this control mode hardware in locked compartments allows maintenance to be performed on most of the processor with little risk of tampering, since no access is possible without supervisory concurrence. This is the recommended type of protection.

HIGH-SPEED MEMORY MODULE

Systems permitting multiprogramming and multiprocessing with both time and space sharing tend to have large capacity high-speed memories. The relative fraction of system cost devoted to memory has increased as this

capacity has increased. Keeping the functions of these memory modules as simple as possible is consistent with system economy, consequently, no special security features are recommended for inclusion in memory modules. Parity checking is presumed as part of normal design on received addresses and data for writing into memory, as is parity regeneration and checking on memory reading. The word size is increased by the 2 flag bits recommended.

Memory bounds registers are not placed in memory modules since as many sets would be required in each memory module as there are in all processor modules and input/output control processors to allow the same degree of flexibility in memory usage. However, the added protection is slight and makes the extra cost unwarranted.

INPUT/OUTPUT CONTROL PROCESSOR

An input/output control processor (IOCP) provides the data flow and control interface between the high-speed memory modules and the peripheral units (either terminal devices in a work station or bulk file controllers). The main function of this interface is to ensure the integrity of the data and the routing between its proper source and destination. Since data rates of the work station equipment are individually much less than the data transfer rate with a high-speed memory, an IOCP can control many peripheral units having concurrent different service demands. An IOCP is therefore a limited capability special purpose processor. Its program control instructions are descriptors prepared by any processor and received via any memory module. IOCP functions for control include receiving, repeating, or generating control signals between or for the peripheral units or one of the processors (executing the input/output part of the ECP). The normal functions include fully buffered independent access with memory modules,

peripheral unit selection, interrupt generation or relaying for processor notification or servicing, and data transfer buffering including both rate and format matching for a variety of types of peripheral units.

The IOCP, by including data buffering for each of the many slow speed inputs or outputs with peripheral units, allows rapid sequential exchange with high-speed memory. This technique is economical since the amount of shared hardware consistent with the secure information interchange is maximized.

Although the IOCP functions could be performed serially on a fast enough general-purpose computer, the hardware provided in most such computers for arithmetic and control operations would be under-utilized while at the same time the computer input/output might be overloaded. The overload would result from the necessity for sequentially completing, in real time, the many asynchronous inputs from peripheral units, without sufficient buffering to smooth the peak demands.

Figure 4 indicates the functional elements contained in the IOCP. As many one-way unit buffers are provided as there are peripheral units. A two-way peripheral unit such as a bulk file controller is assigned two related unit buffers. Each unit buffer connects to only one peripheral unit. A service demand queue permits a steady stream of information flow through unit buffers to be maintained by providing demand priority service for those units whose present contents are nearly exhausted. Program control in response to service demands selects the unit corresponding to the demand and also addresses the data and control words of the I/O memory. The control word provides the appropriate high-speed memory module address and memory bounds on allowed address.

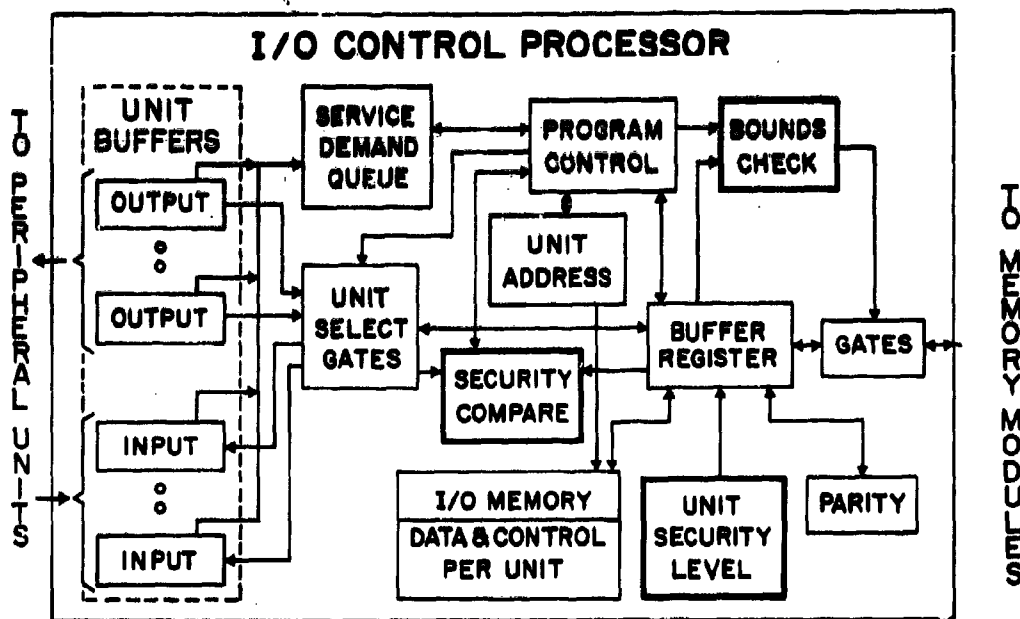


Figure 4. I/O Control Processor

The memory bounds concept is applicable to the IOCP in a simplified form. Simplification results from requiring only one-directional increments of high-speed memory address. This is sufficient, since the processor can order the data as it requires by either increment- or decrement-addressing for successive accesses to memory. The IOCP addresses memory as indicated by descriptors. A descriptor requires the memory block beginning address, and the number of words to be transferred in order to select and control the transfer of arbitrary length records into or from a block of memory dynamically allocated. Addition of the memory block end address to the descriptor provides sufficient information for a memory bounds check.

In operation, at the start of each access the size is checked to be positive. If so, the low address is checked against the end address and then transferred to memory. At confirmation of receipt by memory, the size is diminished by one and the low address is increased by one; both are then written back into the control part of I/O memory for that unit. Only one-way alteration of the low address (increase) and of the size field (decrease) is allowed.

No single addressing hardware error can escape undetected. Should the low address fail to increment when the size reaches zero and the control descriptor is returned to memory, this condition would be detected since the low address would not match the end address. Should the size fail to decrement or start out too large, the end address would be exceeded (memory bounds check) and set off an alarm. Should the low address start with a value too low, the check of control code expected in the first word addressed would generally not match that in the descriptor and would give an error alarm; if it did match, there would most likely not be a security violation (or need-to-know violation). * In any event, this error would be detected when a zero size and a low address different from the end address appear in the control descriptor.

* A match of a random word in memory against a control code would require that every bit of the random word be the same. Control codes could be chosen to avoid common alphabetic words, or even binary coded decimal characters. Thus, the probability of match against a word that is not intended to be the control code could approach 2^{-n} , where n information bits are in the control code. If $n = 48$, this probability is less than 10^{-14} .

Other single failure opportunities are negated by parity generation or check on inter-equipment address and data transfers (with memory and terminal units) and by assigning unit addresses to include parity. These latter are checked on accessing the I/O memory and are included in the decoding and selection gates for the unit buffers. Thus, there are no conditions for single hardware error causing security violation identifiable for the IOCP at this level of its design. Consequently, the information exchanged between peripheral units and high-speed memory is safeguarded by the IOCP, and only processor error can affect security protection of this data in the high-speed memory.

Security comparison circuits are provided for use in several ways. Comparison of the control code in a descriptor available in the I/O memory is made against the control code of a physical record read from bulk file. Comparison is made against a user's key pattern from a user's key pattern terminal unit (refer to work station description, page 51). Comparison is also made to verify the connected unit buffer.

The unit maximum security level allowed by physical security considerations for any information exchange with a peripheral unit is checked against the security part of control codes to assure no release of more highly classified information than is permitted. The maximum unit security level indication is implemented by a wired-in section of memory not changeable under program control. In this form the IOCP provides a redundant hardware security limit check which is independent of any performed by processor hardware or software. This check does not consider the terminal unit user's maximum security clearance or need-to-know and is thus not sufficient for total security control.

All information interchange (data, descriptor, or addresses) should be parity protected against single hardware failure. Alternatively, and at slight reduction in redundancy, the addresses assigned to terminal units within each work station may be grouped so that at least a double bit error is required for an address error which will result in a mistransmission to some other work station; single bit difference is permissible in addresses within a group, since a single bit error would at worst result in another address in the same work station (at least as far as security violation protection is concerned), and since the user could detect the single address bit error by anomalous behavior of the misaddressed terminal unit in his work station.

A security risk inherent in multiple user programs being allocated high-speed memory space is that the residue of the prior user may be read by the newly assigned user. Thus, memory blanking and checking is required as part of the reallocation process. This may be done by software as described in Section IV. In a system which is computation-bound rather than input/output-bound, the memory blanking and checking responsibility can be included in the IOCP to relieve the computation burden.

The IOCP could accept a descriptor which serves to blank a memory block and verify the blanking process. This task is adequately achieved by the storage of a constant value and then reading back the same value for each word in the block. The IOCP has the block addressing and bounds verification capability. It also has a word comparison register normally used for security comparison. This hardware can be readily used to add the memory blanking feature. A special dummy unit buffer is added (connected to no terminal unit) which is used both for input and output. It is initially loaded with a one word output. Thereafter, two descriptors addressing the same block are provided and a sequence of alternate input service

demands followed by output service demands are made by the dummy buffer. Program control provides completion of each word service (in all cases treated as the first word of a block). Thus, the IOCP program control first loads memory, then, on servicing the output part, compares the contents of the dummy register with the contents of the buffer register. Upon establishing equality, indicating that the addressed memory word was blanked, it steps the address to allow the next word to be blanked. The dummy buffer is given lowest priority in the service demand queue and, thus, the memory blanking serves as an effective use of otherwise idle time. At completion of the block blanking and checking, the IOCP returns the I/O-complete descriptor with notification of success. The ECP, at its convenience, returns the blank space to the available space list.

WORK STATION

The work station for a user contains a group of one or more input/output terminal devices plus a user's key pattern generator in a physically secure enclosure. Only one user at a time is allowed in a work station so that the security controls necessary to restrict access to that user (his user's control profile) can be taken as restricting information exchange with terminal devices in the work station. The user's key pattern generator transforms a physical user's key into an electrical signal, the user's key pattern, which is used to initiate user identification-authentication, and to verify continued user presence before allowing classified input or output. At either insertion or withdrawal of the user's key an interrupt is initiated for processing action to change the work station status.

The hardware design of the user's key pattern generator is part of the authentication process and is thus beyond the scope of this study. Its output signal should not be observable or identifiable as belonging to a particular user.

Protection against a hardware failure can be achieved by assigning each key pattern with sufficient distance (bits which differ) from all other allowable key patterns. Thus, if the distance is n , then dropping (or adding) $n-1$ or less bits will be detected as error, and will not be interpreted as another key pattern and, thus, another user.

Shared Use of a Work Station

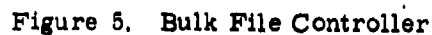
Concurrent use of a work station by several users imposes several complications to the recommended single-user-at-a-time system.

1. The user identification-authentication sequence must be kept private even when performed with others present.
2. Release of security information is restricted to that which is releasable to all users present.
3. A more complex key pattern is required which continuously must indicate whatever combinations of multiple users are present in the work station.
4. No obvious rule exists for limiting the assignment of control code names (indicating the security classification and need-to-know) for information to be added to the data base

In Appendix V, partial solutions to these complications are discussed.

The recommended system with a single user in a work station allows private identification and authentication. The user's control profile corresponding to the identified and authenticated user provides the complete reference for control of release or acceptance of information within the work station.

The bulk file controller serves as an address and buffering device for transfer of a physical record between the actual physical units of the bulk file and the connected input and output unit buffers of the IOCP. Figure 5 indicates its function. It receives control commands for a single access at a time including file addresses relayed from descriptors in the IOCP. It performs parity checks on each word exchange.



53

which is addressed, that it is read in its entirety, and that no alteration of this record has occurred. In form, the sequence indicator should be a combination of the actual file address of the physical record and the contents of this record. The recommended combination is the logical exclusive-or sum of the base address and the actual contents of each word in the physical record. The bulk file controller, in reading a physical record, makes a link check on the sequence indicator to assure that the addressed record is the one read and that the entire record is read without error. An error detected causes an interrupt return to the IOCP; the sequence indicator is not passed to the IOCP in any event. Combination of this block parity with the word parity allows detection of almost all errors. Only the rare combination of an even number of errors occurring in the same places in both dimensions could be missed.

BULK FILE

The bulk file is made up of storage sections which are addressed by the bulk file controller. The file includes means for transferring data blocked into predetermined length physical records to or from the bulk file controller.

Added to each storage section of a bulk file is the capability to physically disable the write circuits. This capability is used to make the content of that storage section safe from alteration. A security feature added to the bulk files is to include a locked compartment within which are the write disabling switches. Also included in this locked compartment is the means for setting the execute-only flag bit for each word of those physical records which contain the executive control program and its associated service routines. This is the only way to set the bit and, thus, identify that the programs are able to be run in control mode.

HARDWARE ADDITIONS FOR SECURITY PROTECTION

Table 2 indicates the estimated additional hardware required in order to implement the recommended hardware security features beyond that required to achieve the normal functions of the modules.

The total for a user's key pattern generator is an estimate of what is necessary for a fairly complex device yielding a 24-bit user's key pattern. Its detailed consideration was not part of this study.

The common measure used for the estimates is the equivalent flip-flop (EFF). An equivalent flip-flop is a group of logical elements having a cost comparable to that of a flip-flop. To provide a basis for comparison, the Burroughs D825 computer module contains approximately 2500 EFF's, and each D825 input/output control module contains approximately 1400 EFF's to service a single peripheral unit.

In summary, the main processor in the future system will operate in two modes, user and control. The control mode has access to the full set of available privileged instructions which establish the area of control over an approved user's program. Entrance to the control mode by a user must always be by means of an established maskable interrupt system. In the user mode, available instructions are limited to the subset not required for security control. These instructions are sufficient for the programs approved for the user. Access to any additional instructions without entering the control mode via an interrupt is impossible. Control hardware is protected from unauthorized access by physical sealing or locking along with suitable alarm protection.

Since keeping memory cost to a minimum is recommended for system economy, no special security features are suggested for the memory modules of the multiprocessor system other than standard parity checking.

Table 2
Hardware Added For Security

<u>MODULE</u>	<u>NUMBER</u>	<u>TOTAL*</u>
PROCESSOR (16-bit bounds)		
Redundant mode control	1	4 EFF's
Memory bounds	3	120 EFF's
Bounds check	3	60 EFF's
I/O PROCESSOR (N terminal units)		
Unit security level	1	20 EFF's
Security compare (48 bit)	1	180 EFF's
Bounds check	1	20 EFF's
I/O memory addition: logic	1	N + 20 EFF's
I/O memory addition: stack	1	N WORDS
WORK STATION		
User's key pattern generator	1	100 EFF's
BULK FILE CONTROLLER (24-bit check)		
Sequence indicator check	1	120 EFF's
BULK FILE (M storage sections)		
Write lockout and Execute only flag bit set	M	2 M Switches + 5 EFF's

*EFF = Equivalent Flip-Flops

The input/output control processor is designed to handle the input and output of peripheral units by individual maximum-security-level-limited buffers to ensure proper routing to intended users and security control. A special bounds-checking capacity established by the main processor further ensures security between the IOCP and its accesses to high-speed memory.

Security techniques for the work stations provide for a user's key pattern generator for identification and restrict operations to a single user at a time in a physically secure area.

Bulk file techniques utilize discrete storage sections to partition different blocks of information. The bulk file controller provides special content checking and verification procedures to guarantee that all of the permissible information is accessed and transferred properly.

Redundancy throughout the hardware design area is reflected in a system of parity checking to verify contents of all addresses and data exchanged between modules, and in security checking hardware repeated in separate modules so that multiple checks are performed for any information release.

SECTION IV

SOFTWARE SECURITY TECHNIQUES

The previous section defined the hardware techniques recommended to achieve fail-safe security. Implicit in the hardware recommendations were several software security techniques which utilize hardware techniques in order to exact the necessary security control. This section describes the recommended software security measures.

The executive control program (ECP) provides the structure within which security routines and techniques are applied; these include the use of specialized control tables, redundant programming, control of user's job execution and programming actions, and system activity logging.

OPERATING SYSTEM AND EXECUTIVE CONTROL PROGRAM

Projected operating systems will be designed on a modular approach utilizing a functionally segmented collection of programs which are dynamically organized as dictated by the system load. An operating system implements and controls the multiprocessing and multiprogramming capabilities of the system and performs the techniques developed for security assurance during electronic data processing of multi-level classified information. The operating system has two main sections: the central section, or ECP, and the service programs section.

The ECP performs some of the security control checks and those control functions normally associated with an operating system, such as established I/O control, interrupt control, timing, etc. The ECP resides in a protected high-speed memory.

A service program is a program written much like a user program but possessing direct contact with portions of the system normally reserved for the ECP. A typical service program is one which can communicate directly with an analyst console at a work station, as opposed to making a call on the ECP to perform the communication. Service programs are protected when in high-speed memory by an execute-only flag bit which allows execution only by a processor in control mode. They have the required built-in security checking techniques peculiar to the type of service being performed.

In Section V, the ECP and service routines will be introduced as the functions are required to control the system and user's programs.

Tables and Directories

The ECP and the service programs provide the user with the action needed at each point in his job processing. A series of tables and directories is required to keep track of user's jobs and the operating system itself. These tables and directories reside in protected memory, are constructed by the ECP, and (except for the program reference table) are available to the ECP or to the service routines only. Representative required tables and directories are identified in the following list:

Equipment availability table - contains status and hardware-imposed maximum security level for data processing for each equipment in the central data processing system.

Channel assignment table - categorizes each I/O channel according to unit type, use, work station, and hardware-imposed maximum security level; determines appropriate I/O service program for channel control.

Inactive job table - contains parameters used for scheduling, accounting, and input/output control for inactive jobs.

Job collection queue - program list for jobs ready to be entered into high-speed memory and executed.

Memory directory - lists the block size and object name, if in use, for each high-speed memory block.

Schedule table - contains a list of all programs which are ready for processor time scheduling, and the name, memory requirements, and scheduling information for each program.

Active job table - basis for control of each job currently being executed; includes state of all registers pertaining to this job at any job interruption; it is used by the ECP for internal processor scheduling.

Program reference table - contains the name and/or descriptor for each object referenced by a program; in addition, it contains a memory-bounds-load flag bit, presence indicator, memory bounds, and base address for each of the objects allocated high-speed memory space. References by a user program to this table are controlled by the PRT memory bounds register.

System directory - lists all files currently being used by the system; contains the name, control code, size, and location of the file directories for each active file.

Inactive file directory - lists all files available to the system but not currently in use.

File directory - contains information pertaining to identification of each object in a file such as name, location, control code, and size.

Special Tables

In addition to the above tables which are part of basic ECP control, three other tables are provided explicitly for security control. They are the control code name table, the system user table, and a set of user's control profiles, one per user.

Control Code Name Table

The control code name table provides information for conversion between control code and control code name. There are two ordered parts, with the first part being ordered by control code. Associated with each control code is a pointer indicating the address of the start of the variable length control code name. The second part is ordered by control code name. Associated with each control code name is its size and the address of the control code. Thus, corresponding entries in each part are address linked. This table design depends upon addresses to entries within this table being significantly shorter than the control codes or control code name.

System User Table

The system user table provides the file name or high-speed memory address of each user's control profile. This table is ordered by user's key pattern. Each entry has an "open" indicator which, if set, signifies that the profile is activated, (i. e., the user's key is presently in a key pattern generator at a work station and the user's key pattern has been authenticated). An indicator of presence in high-speed memory and an address pointing to the location of the user's control profile are included. Also indicated is the alternative identification field to be used only by the responsible supervisor when profile updating is required. Because of the relatively few system users and infrequent updating, location by means of this unordered secondary identification is no serious penalty.

User's Control Profile

A unique user's control profile exists for each user of the system. It provides the information necessary to associate to that user the security classifications and need-to-know authorization deemed necessary by his supervisor for effective performance of assignments. Each user's control profile contains three parts: header, user authentication information, and control code list. The header includes the control code of the profile itself, the user identifier (an image of the user's key pattern), and size and self-relative addresses for the beginning entries for the other parts of the profile. The user authentication information provides the response information expected from the user to validate that he is, in fact, the user indicated by the user and key patterns transmitted. The control code list is separated into two ordered control code parts. The first part contains all control codes for which both

read and write access is permitted to either the data base or terminal units in the user work station. The second part contains all control codes for which read access only is permitted from the data base. These can be output to the terminal units.

The user's control profile is in a read-restricted and write-protected area of memory. Reading is restricted to an ECP-called I/O service program. Alteration of any user's control profile is done only by an ECP-called service program which has its access protected by a control code unique to the system supervisor. This program is usable only by that user's supervisor, who must also gain system access through an identification-authentication procedure, and who has write access to a control code reserved for his use only. If this degree of access control is inadequate, it can be coupled with a properly identified second person (possibly the user himself) who must concur in the supervisor's action.

EQUIVALENCE PRESERVING TRANSFORMATION

For a user to bypass protection techniques added to his program by the ECP he must know how his program is to operate. It may be possible to develop a program which converts user programs to "equivalent" programs by "equivalence preserving transformation". This technique could be an extension of the work by J. Nievergelt,* which proposes a program to examine any other program and perform such simplifications on it as can be detected by the "argument" program's form alone, without having any knowledge of what it is supposed to do.

*Nievergelt, J., "On the Automatic Simplification of Computer Programs," Communications of the ACM, Vol. 8, No. 6 June 1965, pp. 366-370.

The application to security protection would include as a first step the separation of data from programs so that the program itself (instructions and literals) is unmodifiable during execution (address alteration by indexing of instructions other than jump instructions should be allowed). When the program is so separated, the program can be protected by bounds allowing read only access to the user. In a mixed content stream (program instructions, literals, and variables), recognition of which words contain instructions versus constraints is not possible from the words alone unless flag bits are included. The compiler output is the proper place to tag words as instruction, literals, or as data (differently tagged). Applied as a post-compiler or assembler to a syntactically correct compiler or assembly output, the equivalence generating program could develop several equivalent versions and select that one which has the most desirable security characteristics. If re-compiling were done as part of compile-and-go operations and if some randomization criteria were used on equivalent program segment selection, the size of the program would vary and with it the order and location of inserted security checks would change.

REDUNDANT PROGRAMMING

Redundant programming may be defined simply as follows: to achieve results by differently programmed means and to ascertain that the results are equivalent.

Redundant programming can be a time-consuming means of testing reliability if it is not designed into the hardware in the first place. This does not exclude the possibility of utilizing redundant programming techniques at key points in a routine for confidence testing, however, it does limit the concept of total program redundancy for the system.

Operating the same program in different modules (this presumes at least two modules of each type) at least doubles operating time since a means to verify the results is also required. Utilizing different instructions to derive the results would more than double the operating time. This assumes that the original routine used the most direct (fastest) instructions; the alternative routine instructions would take longer. Verifying that the results are the same adds an additional time load.

Often, given the results of a program, an inverse program can approximately reconstruct the inputs. In some cases this inverse program is so short compared to the needed program that its use adds little to the total execution time. The disadvantage of this method is that it is only of limited application.

One method of redundant programming in a multiprocessing system (with more than one computing module) which would not increase the overall operating time or cost is described by the following example. Assume that some processor module, say C, is used by the ECP to ascertain that a user's information retrieval request is within the scope of allowable requests for the particular user. C then initiates the input request to read a file or record from the bulk file into a high-speed memory block. The appropriate ECP system routine executed by C initiates the input action and masks C from responding to the action termination. Upon input termination, the ECP input confirmation service program is activated in the interrupted processor module (any one other than C) where the check is performed with the control code of the data record and the user's control profile. If the check is satisfactorily completed, the data is released to the user; if not, an appropriate alarm is given. The danger of releasing unauthorized data with this type of redundant programming technique used by the ECP is low, since multiple hardware

malfunctions would have to occur prior to such release. Checking procedures are serviced by different processors operating in control mode, and the memory is protected by parity checks and bound registers associated with the user program and taken from his program references table; this process is described in detail in the next section.

CONTROL OF USER PRODUCTION JOBS

All requests for programs and data must be processed by the ECP. At the time of job production scheduling for a user (not program preparation scheduling), the user's job is assigned and a skeleton program reference table (PRT) for the requested job is assigned to a PRT area. This skeleton is complete except for actual locations and the presence indication for each entry. The ECP verifies the right of the user to have access to the desired information program segments and data blocks named in the PRT by checking the control code of the named information in the system directory against that user's control profile. If the request is approved, the ECP or ECP-called service programs allocate memory space, enter the assigned base location in the PRT and, along with the computed upper limit for each named PRT entry, obtain the required programs and data from the bulk or other storage, and place these information entities within the high-speed memory. The ECP program then checks the header contents of each retrieved data block for control code against the expected control code (and possibly data block name or block number). If this check is satisfactory, the corresponding PRT entry is marked as present in high-speed memory. Some or all of the memory bounds registers may also be set by the ECP at this time. The computer, after verifying that everything has been set up correctly, initiates the user program.

Relative addressing from a base register (or index register) can be accomplished provided the associated memory bounds register has been properly set. Base and index registers can be changed by the request of a user program (in user mode) and new areas are thus opened up for writing and/or reading, provided the new register values come from an indirect address reference to the PRT area and the referenced word has the memory-bounds-load flag bit.

Indirect addressing via the program reference table is possible, combinations of the various floating techniques are also possible, e.g., indirect addressed transfer, via the PRT, to a subroutine with addressing relative to its own base. Any attempt, accidental or deliberate, by the operational program to address (read from or write into) any area of memory not specifically permitted by bounds registers established by the ECP is immediately detected, and the processor is returned to control mode for ECP interpretation. There is, likewise, no way for the user program to use any I/O device without calling on the ECP. Thus, excepting hardware failures, such a system should provide absolute protection.

Maintaining this protection in the event of any single-failure situation is a subject for special study of particular hardware. The recommended hardware features described in Section III provide sufficient redundancy to protect against the loss of protection due to most, if not all, single hardware failures. The ECP itself is capable of utilizing redundancy to verify its own results by executing the program in different processors in a manner similar to the redundant process described on the previous page for checking input from the bulk file. This will be achieved in normal operation since the work load is shared among all processors without reservation of any one for particular tasks. (Although a particular interrupt may be

directed to a particular processor, the assignment need not be made for more than one interrupt; the next time, a different processor may be made responsible.)

CONTROL OF USER PROGRAMMING ACTION

For a user to illegally manipulate the modular, multiprogramming, multi-processing system, by his programming action, he must:

1. know its operation, organization and security protection techniques,
2. gain access to the operating system while in process of controlling this system, and
3. recognize and successfully avoid execution of all sub-routines which can detect, log, or abort his manipulation.

Protection can be provided to prohibit a user's illegal manipulation of the system by the following:

The operation, organization, and security protection techniques can be divided into n parts. These parts are assigned to different personnel for implementation so that any one person only knows a small part of the entire system. Any attempt by an individual to gain knowledge of a part of the system that he is not responsible for can be readily exposed and becomes a matter for administrative action.

Gaining access to the operating system can be prohibited by making memory bounds registers fail-safe (the program does not depend upon the bounds registers for limiting addresses; thus, bounds transgression can and does cause interrupt) and by wiring in (through locked switch permitting writing on a bulk file) the access to the ECP so that the only way to alter the system is through a hardware interrupt (caused by either hardware or software) which leads to a specific interrupt address for initiation of ECP control.

Random insertion of program challenges in a program string forces either knowledge of the proper responses (as would be available to an authorized user of the program) or interpretation and avoidance of such segments of a program string. In the latter case, the program must be treated as data for interpretation before it is executed. Tagging each memory word used for a program differently from those used for data would prevent any attempt by a user program to analyze another program as if it were data. The provision for monitoring of undebugged programs or their interpretation by a service program (assembler or compiler post-processor) may still be allowed as an aid to programming, so long as the output error messages or monitored results are related to the source symbolic language rather than the internal machine language.

The block concept used in ALGOL 60 provides inherent program security protection. In this concept, programs are segmented into blocks. External entry into a block is only at its block head. Processing from the block head continues until eventual exit from the block returns control to another block (or completion of job). The block heading may include parameter declarations which provide linkage to a local block. Any such transfer into or out of a block represents a potential program breakpoint at which security check procedures could be executed without effect on the operation of the actual program so long as no names in the user programs are set equivalent to locations in the check procedures.

SYSTEM ACTIVITY LOGGING

Logging is a controversial subject mentioned only to exhibit the after-the-fact security breach detection inherent in a system containing this feature. What logging is required is an administrative decision. Assuming that a system log is required, it should be a service program which keeps a running log on

each user's requests and the responses to these requests from the data base. This log constitutes an historical accountability file of user activity. This file is used to determine activity rates of various users and, more importantly, the activity record for information within the data base. The historical files, when used to measure personal performance, can detect inefficient procedures as well as behavior patterns inconsistent with secured analyses. Analyst knowledge of this use of historical files tends to minimize unprofitable requests and make penetration attempts more difficult. The control code assigned to a log file would be one available only to supervisors. Log information can be separated into multiple records with different control codes available only to different supervisors. This diffuses the analysis responsibility among multiple supervisors so that cross checking can be achieved.

In summary, the software security techniques recommended in this section for application to the modular, multiprogramming, multiprocessor system defined in Section III fall essentially into five areas: the executive program control, tables and directories, redundant programming, user production job controls, and prevention of illegal manipulation by system users.

Projected modular systems will utilize a functionally segmented collection of programs which are dynamically arranged to fit the system's load requirements and which are under control of the executive control program or an ECP-called service program. These programs provide the required security checking, scheduling, and bounds limitations to prevent security compromises.

To keep user's requests and the entire system orderly, a series of tables and directories is established in protected memory to be called on as necessary by the ECP and service routines. Within this series are three tables especially constructed for security control: (1) the control code name table

which ties together control codes and control code names, (2) the system user table to locate each user's control profile in the high-speed memory, and (3) the user's control profile itself, which not only identifies the user to the computation complex but establishes his need-to-know and security levels for the system.

Cost considerations recommend redundant programming be used only for security checking and be accomplished by at least two different paths, both to verify results and to assure that security is maintained.

All requests for programs and data must be processed by the ECP. This user job control assigns table reference, order of processing, performs validity checks upon the user's right to information and to area bounds, and subsequently checks the requested data itself to insure its availability. The ECP sets memory bounds as necessary and releases the computer to start the user's program. Relative and indirect addressing are also possible within preset bounds.

Techniques to prevent illegal manipulation of the system involve (1) restricting an operator's knowledge of the system to just the one area he is responsible for, (2) fail-safe memory bounds registers, (3) insertion of random program challenges which can be answered properly by an accredited user only, and (4) the ALGOL 60 block concept.

SECTION V

ECP APPLICATION TO CONTROL USER PROCESSING

The previous two sections have described and discussed various attributes of several hardware and software techniques along with recommendations as to which can provide fail-safe security protection for an advanced EDP system, as defined in Section II.

In this section, the recommended techniques for security control are integrated and applied to system startup and use. The discussion focuses upon the controls provided by the executive control program (ECP), the special security routines, and demonstrates the use of hardware techniques. Control of the multiprogramming, multiprocessing system defined in Section II is provided by the ECP, using the recommended hardware techniques described in Section III and the tables and directories and other software features described in Section IV.

Security protection provided by the ECP must be fail-safe. Fail-safe security protection means that information processing must be interrupted by (1) any detected hardware malfunction, or (2) any programmed attempt to transgress permissible security bounds (or alter them in any unauthorized manner). In other words, any suspicious action results in cessation of job processing until the suspicion is either borne out or proved groundless. Only in the latter case is job processing continued. Fail-safe implementation requires that at the completion of each major information processing

step positive action must be taken to permit initiation of the succeeding step. Each positive action is a specific check which must yield a corresponding specified response; otherwise the process is interrupted, and the failure (presumed) or forbidden action request is investigated.

Specification of appropriate responses to failed checks is the prerogative of the particular system design. Responses can include completely programmed reactions such as logging or calling in of diagnostic service programs. They can include output for administrative action, or they can either notify the user of an inability to continue or stall the user while administrative notification is given.

The requirement for security protection in the presence of a single hardware failure is satisfied in large measure by fail-safe implementation of error-detecting circuits at those points in the system where processing of security checks is performed. Such methods include parity, illegal address detection, memory bounds checks, flag bit presence, etc. Appendix IV reviews achievements of arbitrary reliability by hardware methods. Subsequent to detection, programmed diagnosis of the cause and scope of the error is required to determine necessary corrective action. Following repair, a programmed confidence check must be passed before the repaired equipment can be returned to normal operation.

Safeguarding security of information in an EDP system also requires protection of the security controls placed on the system as well as protection of the classified information itself throughout its input, processing, storage, retrieval, and output. Figure 6 is a summary flow chart showing these controls during user job flow. During system startup the controls are established by the system supervisor; after operations begin, the ECP

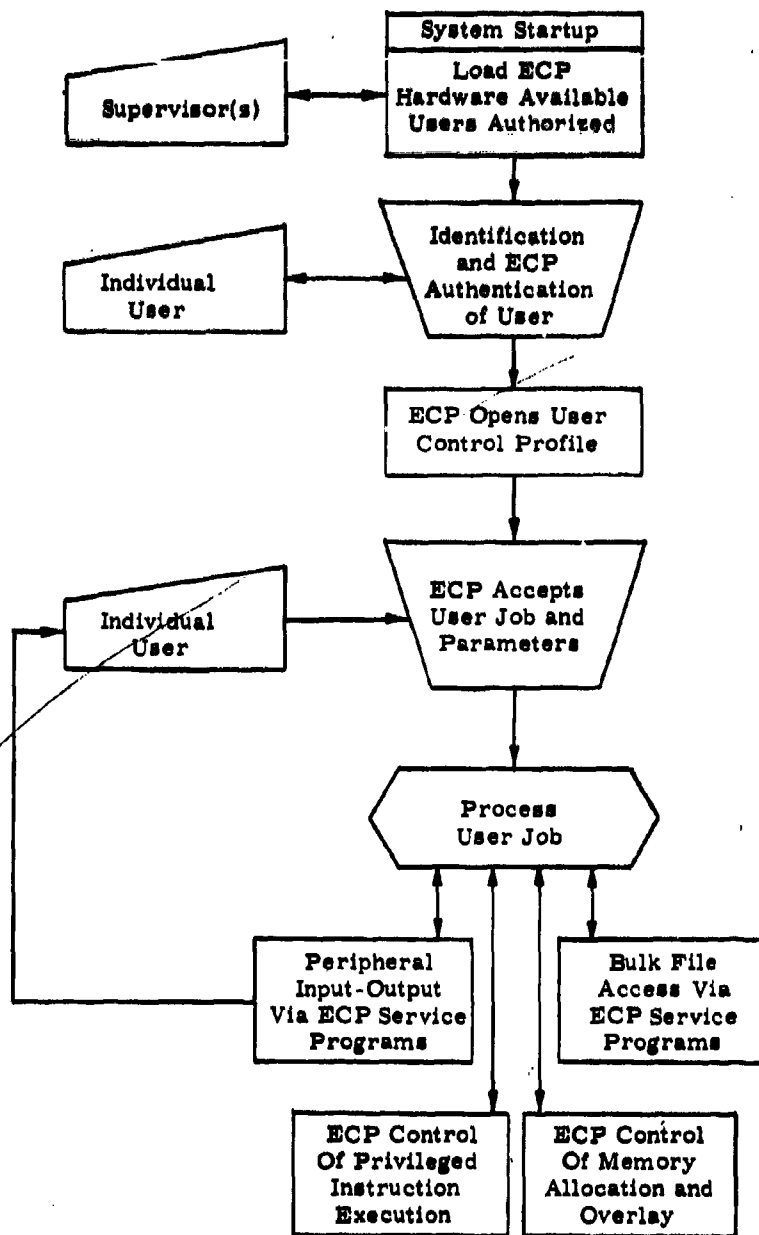


Figure 6. User Job Flow Control by ECP

takes control and carries out the details of safeguarding software security. Any supervisor-authorized individual user may identify himself to the system as user of a work station. When satisfactorily authenticated, he may enter or have access to classified information as limited by his user's control profile. Control by the ECP over user jobs includes input/output, privileged instruction execution, memory usage, and bulk file access.

Any undetected error in input control code name assignment to a record by a user may result in unauthorized downgrading and a security (need-to-know) violation. If the error results in upgrading or need-to-know change it may result in the information becoming inaccessible to authorized users. Violations may also result from processing errors such as (1) conversion of control code name to control code or the converse, (2) control code alteration or disassociation from the record of the control code which describes the record security classification, and (3) need-to-know restriction. Undetected input or a processing error can lead to violations through mis-storage of the record or proper storage of a misclassified record in the data base. Any retrieval of such a record thereafter is a potential violation, as is any error in the check made on the control code of the retrieved record. The actual violation occurs on disclosure of security information to a user without sufficient clearance or need-to-know. A control against such violations depends upon the reliability of the information against which the checks are performed. Thus, if the control code used to protect a record is improper as a result of undetected errors made in the input, processing, storage, or retrieval, the proper information for safeguarding the output may not exist. It is evident then that security safeguarding must consider the entire EDP process, even though the actual violations occur on output.

STARTUP

The system startup includes loading the ECP, establishing the available equipment configuration, forming user authorization tables, and releasing the system under ECP control (for processing of user jobs).

Loading ECP

Loading the system initiation portions of the ECP is required to start the system.

A wired-in program loader is available which first automatically clears all memories and registers at startup and then establishes memory bounds for the input area into which the ECP is to be loaded. Normally loading is from pre-assigned sections of the bulk file containing the ECP. An alternative loader from some other input device is required for first-time loading and later as backup. The pre-assigned sections of bulk file containing the ECP have positive write-lock-out protection (alterable by supervisor only).

In a multiprocessing configuration, two different areas of high-speed memory are assigned for ECP loading, after which the loaded copies are checked against one another. Upon satisfaction that one version is properly loaded, the write circuits to that memory module could be disabled (at least the external connection to them in destructive read memory systems), and the other copy can be overwritten under control of the first. Thereafter, so long as that first area of memory is operating, the ECP is unalterable by any electrical means and is immune to user program actions. If an error should be sensed in the ECP area of memory, the fail-safe approach is to repeat the ECP loading procedure into a different memory module. User programs in process might have to be aborted and later repeated from a convenient recovery point. With enough modularity, and if the need for

uninterrupted performance should dominate the fail-safe need, two copies of the ECP resident program could be kept in high-speed memory. An error from the memory module containing an ECP copy would divert ECP action to the other copy which would take the suspicious module off line, and the system would continue after a check for error effects. In any event, the security protection potential inherent in a user-unalterable ECP is not compromisable so long as the ECP cannot be bypassed.

Establishing the Available Equipment Configuration

After the ECP has been successfully loaded, it checks to see what physical equipment is currently operable and available to it. This check includes all the physical equipments (memory modules, processor modules, I/O control processors and associated terminal devices, bulk file controllers and the files themselves, etc.) to see which are available to the system.

The startup availability of an equipment includes confidence tests on the equipment such as leapfrog tests for a high-speed memory module (a test which eventually occupies every possible position in the memory). These confidence tests are performed on the total system at startup and are repeated on individual system modules at appropriate intervals during the running of the system to ensure continued confidence in its operation.

Successful completion of a confidence test opens the entry for that equipment in the equipment availability table. This table entry contains the equipment maximum allowable security level as limited by the physical security of the location, the electromagnetic radiation characteristics of the equipment, or the routing of interconnecting cabling. This table is prepared by a service program using supervisory inputs. The permitted security level of the equipment being used is checked with the security part of the control code of all machine records prior to implementing any

I/O call. This check is redundant to the maximum security level check performed by the IOCP hardware. The software check simplifies temporary reduction by a supervisor of the maximum security level allowed a peripheral unit which might be necessary during maintenance. Alternatively, raising the security level requires two changes (by different supervisors, if necessary), one in the locked section of the IOCP and one in the equipment availability table.

Forming System User Table

The ECP will now accept supervisory interrupt from a peripheral device to enable setting up the user's control profile tables for other currently allowable user's (each containing the control codes for all authorized security classification and need-to-know categories by access permission, as well as identification-authentication means).

The originally loaded ECP contains a routine which can only identify supervisors. Therefore, the only initial interrupt the ECP will recognize and service is from a supervisor. This routine may be supervisor-alterable, use-to-use, as required.

The ECP response to the initial supervisory interrupt is to initiate the authentication sequence corresponding to that supervisor's key pattern. Satisfactory completion of this authentication by the supervisor opens the supervisor's user control profile. That supervisor enters or activates entries in the system user table of identification authorizations for those users which he supervises (as allowed by his control profile) and who currently require system access. Thus, the system user table loading uses a routine available to a supervisor only (i. e., the routine has a control code present only in a supervisor's control profile). The ECP sets up

bounds for this table in a protected memory area (accessible only to the ECP or its assigned service routines). Input is protected through use of the IOCP (see Sections III and VI). At completion, the supervisor receives an output confirming the users whom he has currently authorized, and a log record is made of this action.

Releasing the System Under ECP Control

When the system user table has been completely loaded, the supervisor will release the ECP and effectively open the system to the other users now authorized.

USING THE SYSTEM

With the system released under ECP control, multiprogramming, multiprocessing of user jobs may begin. The ability of the system to perform many operations simultaneously for a single user or for a group of users requires maintaining the integrity of individual user programs, their data areas, and their work areas. At any given time there will be activity instigated by certain user programs while other user programs will be passive in memory.* The evident solution for maintaining the integrity of all users currently in the

*The multiprogramming-multiprocessing capability makes this multi-job presence in high-speed memory desirable to permit much less frequent computer-bound or input/output-bound processing. This is achieved by having a backlog of work of each type available for execution. Thus, the ECP, being responsible for efficient equipment usage, schedules user jobs among available equipments. This scheduling recognizes job interrupts resulting from current unavailability of data or programs in high-speed memory prerequisite for further processing and initiates the processing necessary for accessing it. When the delay external to the processor module in accessing the information is significant (for example, a human response from a work station, a card read, or a line print), that job can be made passive and another job active until after all prerequisites for continuing are fulfilled.

system is to restrict each "currently running" user only to his permissible programs, data, and work areas. The other users at their turn for execution will likewise be restricted to their own programs, data, and work areas, and the integrity will be maintained by self-exclusions.

Each user program may call on any common service program so long as the control code of the service program is in each calling user's control profile.

More than one user program may execute a service program from a single copy of it in a high-speed memory area so long as it is not alterable during execution. Data and working storage areas required by the service program are separately assigned for each calling user program. At call time for the service program, memory bounds are assigned for that user program around the three types of areas involved, and execution is initiated. At completion, release of the results and return to the calling user program is achieved in control mode.

The following thin- or single-thread analysis of system use begins at first response to a user starting to use a work station and continues throughout the processing of his job. The job is assumed to require input and output and to call upon the filing system as part of an information storage and retrieval application. Job termination action completes this flow. Table 3 summarizes the analysis by indicating the ECP or service routines required to control the user program. These routines are given in their single-thread order of occurrence by name, description, and estimated size (in number of instructions). Routines indicated by asterisks are added especially for security protection. Indentation is used to indicate a subordinate security routine called by the basic service routine. Routines enclosed in parenthesis are repeats of previously described routines. Only differences characteristic of the place of occurrence are noted in the summary descriptions given in the

Table 3. ECP Service and Security Routines for User Program Control

<u>Routines</u>	<u>Description</u>	<u>Instructions</u>
<u>Interrupt Processing</u>	Determines cause of interrupt; furnishes required routine for servicing interrupt; initial user identification through system user table, user's control profile into high-speed memory.	400
*Select User's Control Profile	Selects the correct user's control profile for a user.	20
*Activate User's Control Profile	Authenticates user; if OK, opens user's control profile.	100
<u>Memory Allocation</u>	Sets up bounds and assigns memory area for user input.	400
*Check Memory Bounds	Sets memory bounds registers for user allocated area; verifies that bounds work.	50
<u>I/O Processing</u>	Furnishes input services such as resolving conflicts, preparing required control descriptors for IOCP; varies as terminal device and input or output.	1800
*Insert Security Fields	Loads user's key pattern and redundant memory bounds in IOCP descriptor.	20
<u>(Interrupt Processing)</u>	Initiates following subroutines at input complete.	
*Check User at Input Complete	Verifies that the originating user is the same as user at input complete.	10
*Converters	Converts record control code name to control code, or the converse, using the control code name table.	80
(*Select User's Control Profile)		
*Check Control Code	Verifies that record control code is in the user control profile before release for user.	20
<u>Input Responder</u>	Interprets the job request requirements into an inactive job table.	1000

* Added for security protection.

Indented routines are subordinate routines called by the basic service routines. Routines in parentheses have been defined previously in the table.

Table 3. ECP Service and Security Routines for User Program Control (Cont'd)

<u>Routines</u>	<u>Description</u>	<u>Instructions</u>
Scheduling	Schedules collection program for servicing this job.	600
Collection	Assures availability of necessary programs and files in system, prior to job scheduling.	1500
(Scheduling)	Schedules run sequence of jobs from inactive job tables by priority, equipment availability, and high-speed memory space requirements; enters job-required file directory items into system directory.	
(Memory Allocation)	Sets up and assigns memory area for active job table, user PRT, program, files, and work areas.	
(*Check Memory Bounds)		
Readier	Completes active job table for program so that processor control can be transferred to it; calls Find for each PRT entry of a scheduled program.	100
(*Select User's Control Profile)		
Find	Enters the memory location and bounds of the named object in a PRT entry. Calls Locator if object is not in main memory. Returns control to Readier.	200
(*Check Control Code)		
Locator	Locates where a required named object is, initiates the necessary memory allocation and I/O processing. Returns control to Find.	300
Executes Requested Program	Performs function requested by user. (Calls on the following: additional service routines are made as required.)	

* Added for security protection.

Indented routines are subordinate routines called by the basic service routines. Routines in parentheses have been defined previously in the table.

Table 3. ECP Service and Security Routines for User Program Control (Cont'd)

<u>Routines</u>	<u>Description</u>	<u>Instructions</u>
(I/O Processing)	Furnish input/output services.	
(*Insert Security Fields)		
(*Select User's Control Profile)	Affixes description with control code, key pattern, and bounds.	
(*Check Control Code)		
(*Converters)		
Filing System		
	Enters new files in file directory and enters file directory information in system directory.	9000
(*Select User's Control Profile)		
(*Check Control Code)		
Terminate		
*Memory Overwrite	Performs system accounting and logging.	400
	Blanks out memory after release by user and checks to see that it was blanked out.	20
*ECP Loader		
	Loads ECP in duplicate; compares to assure proper loading.	10
*Create or Change User's Control Profile	Sets up or alters the user's control profile.	100
	Changes required user response to the authentication routine.	50
*Create or Change User's Authentication Data		
*User Current Security Authorization	Outputs the user security authorization to the Supervisor.	80
*Deactivate User's Control Profile	Marks the user's control profile closed to the system.	25

* Added for security protection.

Indented routines are subordinate routines called by the basic service routines. Routines in parentheses have been defined previously in the table.

second column. The first time a routine is listed, the estimated number of instructions is given. The estimates for the basic ECP and service routines are developed from Burroughs experience on the D825 and B5000/5500 data processing systems and on projections from the B5500 system. The estimates for the security routines are developed more fully in Appendix I, where each description includes inputs, outputs, and logical flow diagram.

Initial User Identification and Authentication

Control of a user by the ECP starts when a user inserts his key in a key pattern generator at a work station. Insertion causes an interrupt, identified by unit, which is passed through the IOCP servicing that work station (and all its included terminal units). The interrupt switches the responsible processor (as indicated by its mask register) into control mode and thence to the common starting location for interrupt processing. The user's key pattern from the terminal unit causing the interrupt is used to relate the user to the work station and its included terminal units and to identify the user's control profile. This is done using the Select User's Control Profile security routine which makes the identification by referring to the system user table for the file address of that user's control profile. When the profile is available in high-speed memory (retrieved by the information storage and retrieval system), the Activate User's Control Profile security routine matches authentication responses from the user against the expected responses in the user's control profile. Successful matching activates the user's control profile, and an "open" indication is made in the system user table.

An opened user's control profile provides the control codes authorized for processing information for the user at that work station.

User Interrupt to Enter Processing Input

The identified and authenticated user initiates a processing request from his work station by a suitable interrupt. The IOCP forwards this interrupt, now identified as to unit and user's key pattern, to the processor module responsible for this interrupt.

An input buffer area is required so that the user can identify what job he wishes performed. A memory block is allocated by the Memory Allocation service routine and memory bounds are set about this block and checked using the Check Memory Bounds security routine. An input descriptor is prepared for this block including block bounds as part of the I/O Processing routine. It calls the Insert Security Fields security routine which enters the proper user's key pattern. The descriptor is sent to the IOCP for a specified terminal unit of the work station from which the key pattern was received. The IOCP conducts the input operation and at input completion gives an interrupt. Transmission to the proper area is assured by bounds registers. Parity checking of all data transfers provides confidence in delivery of unaltered data to the assigned unit.

The ECP receives a termination interrupt from IOCP when each input record has been completed. The termination interrupt is serviced by the Interrupt Processing service program of the ECP, executable in control mode only. The Interrupt Processing routine at input complete uses four security routines. The Check User at Input Complete security routine confirms that no different user is now at the work station. This is a redundant check since a prior interrupt should have been received and processed at removal of the prior user's key, and only one key at a time can be used in a key pattern generator. The Converter performs control code name conversion to control code since all internal machine records require a control code, which is, in this case, converted from the control code name included in the user's entry.

After the Select User's Control Profile security routine accesses the user's control profile, the Check Control Code security routine assures presence of this control code in the control profile of the user providing the input, using the key pattern from the source work station as reference. The input assumed here (as interpreted by the Input Responder routine) is a new job request which is interpreted and an inactive job table entry is prepared, including user and job identification.

The Scheduling service routine is now called by the ECP to schedule (in this case) the Collection service routine for preparing this job. The Collection routine checks all of the job requirements and assures the availability of these requirements to the system.

The Scheduling service routine is again called to schedule the run sequence of jobs from the inactive job table by priority, subject to constraints on equipment availability and high-speed memory space. It also enters the job-required file directory item into the system directory.

When a job is scheduled to run, all necessary objects listed in the PRT of the job must be brought into high-speed memory. There are four service routines involved in the process which are called by the ECP as often as necessary, depending upon the location and requirement for simultaneous presence of these objects. The Memory Allocation service routine assigns high-speed memory areas to the active job table (which provides the conditions for registers in order to start or resume the job), the user's control profile, and the PRT for the job; this routine also maintains the memory directory.

The Reader service program is responsible for completing the active job table and the PRT for the job. In order to do this it calls upon the Select User's Control Profile security routine and makes the user's control

profile available in high-speed memory. The Find service routine is called by the Reader routine for each object entry in the PRT. If the control code is there, the memory base location and bounds are entered in the PRT entry for the object. The Find routine first consults the memory directory to determine if the entry is already in high-speed memory. If so, and private use is not indicated, the control code is checked by the Check Control Code security routine, which compares the control code of each object found against the user's control profile. If private use is indicated or if the object is not in high-speed memory, the Locator service routine is called by the Find routine to search the system directory and possibly the file directory to determine the necessary size of the object. The Find routine also calls the Memory Allocation routine to allocate memory space for the object, and initiate the retrieval process. When the desired object is in the high-speed memory, the Find routine performs the control code check and PRT entry completion. When all entries in the PRT have been completed, the Find routine returns control to the Reader routine which completes the active job table by entering the initial register contents for job initiation.

Processing User Program

Control is now passed from the Reader routine to the ECP. The ECP establishes PRT memory bounds, initializes the processor according to the active job table, returns to user mode, and transfers control to the start of the program required to execute the user's request.

References to program data, and working storage blocks are handled indirectly through that user's PRT. Thus, the pre-established memory bounds (including allowed use) contained in the PRT entries are available for reloading memory bounds registers prior to any user program addressing into a block. Flag bit protection of such a PRT entry assures that the content can only be used for bounds re-establishment.

The automatic hardware instruction address comparison against the memory bounds registers confines instruction execution to the user's allocated execute-only area(s). Data or working storage areas may be read-only or both read-and-write access protected by memory bounds registers for the user, as determined by use indicators in the memory bounds registers.

Input/Output Processing

All I/O requests are directed to the appropriate I/O service routine executable in control mode only. For each such request, this program prepares a descriptor for the IOCP which carries out the I/O routine in the secure manner described at length in Section VI. In this discussion, I/O is considered as viewed from the ECP and high-speed memory. An area of this memory receiving input remains unavailable to the user until its control code is checked by the Check Control Code security routine against the user's control profile after the user's control profile has been selected. Once this area is released to a user program, processing of its contents should be unrestricted until an output attempt is made. Control code alteration (through its control code name) is necessary, for example, in the legitimate execution of the user reclassification function. To prohibit this alteration restricts the user unnecessarily.* A reasonable limitation is that a user may alter the control code provided the new control code is present in his output control profile.

* Given two records with different control codes, there is no unique implied control code for a record combining the information. Not even identifying a single control code relating the two others is adequate since this new control code is not applicable for all possible information combinations from the source records. Consequently, the assignment of a control code name to a combination record must be a user prerogative, limited only by those allowed for him by his user's control profile.

There are ways of prohibiting access to the control code associated with data, such as rigidly formatting the data or maintaining the control code in a user read-only section of high-speed memory. Since a user's program must be able to merge data records protected by different control codes, it could interchange the data without altering the control codes and return the record to the data base, effectively achieving control code alteration. Therefore, prohibiting alteration of record control codes is not a fail-safe means of protection.

At the time for output, the user's (output) control profile will determine the acceptability of the output request. This includes ensuring that the control code placed on the output record is within the user's output control code list.

The information storage and retrieval system design is beyond the scope of this study. It represents the structure of the data base content. The software providing access to the data base is, to a large extent, independent of the data base structure. Prior to discussion of this software, some security-imposed considerations in the design of the information storage and retrieval system should be noted which aid in efficient processing.

Access to a named file (containing either program or data) requires knowledge of the file's physical address in bulk storage. A system directory, available only in control mode, provides this name-address relation. The system directory may also contain the control code of the file to allow redundant checking.

File access limitation can be controlled in either of two ways: by checking the control code of the file against the calling user control profile, or by checking the file-user list included in file for presence of the calling user. The first approach is recommended, since it allows examination of a single, fixed-size entry (the control code). The second approach, in general, requires

entries for a variable number of users, the number of which probably changes more often than the control code. Since the check on access should be made prior to granting access, the fixed position single control code is easier to locate and use.

Execution of the I/O Processing service routine prepares and furnishes I/O descriptor information to the IOCP (which may be transmitted on to the bulk file controller). The following discussions treat I/O requests affecting the data base separately from the I/O requests involving the other types of terminal equipment (installed in work stations for user I/O). These discussions are software centered about the processing done by the ECP and service routines for input and output. Complementary discussions centered about the IOCP appear in Section VI.

Input Processing from the Data Base

Depending upon the level at which information retrieval information is entered, requests on the data base for program or data records have several types of checks imposed.

The first level may be on the file index keywords used (a user could be restricted to certain keywords or classes of keywords which are defined by control codes in the user's control profile). This would exclude any access of information outside the scope of these keywords. This level of control is not generally recommended since the information partitioning by keywords is not compatible with the partitioning by control codes representing need-to-know and security classification.

The next level of check is on the retrieval identifier (or file address) of each information record described by the keyword. This requires the identifier (or address of the information so a check can be made against the user's control profile to see if the retrieval is allowable.

The two levels of checks just discussed are made on items that are separate from the actual data that is being protected. If either of these checks fails, access to the information is not permitted. This is accomplished in each case by a fail-safe design that requires proper steps to be taken (positive action) before continuation is permitted. The retrieval system, once it has the address of an object file, accesses the data base (bulk file) through the IOCP and bulk file controller for the required information record.

At I/O complete interrupt, the information in response to the request is in high-speed memory. A hardware memory bounds check in the IOCP on the transfer addresses assures that it is in its assigned input area. Prior to release of this information to the user's program, the access is logged and a check is made that the control code contained in the header of the information itself is in the user's control profile. Another check is made to ensure that the information control code matches the control code associated with the address contained in the information retrieval system directory or file directory. Any hardware malfunction altering part of these would be detected, and any undesired information release would be prevented.

Output Processing to the Data Base

Output to the data base in the bulk file first compares the control code of the data with the user's control profile to see if the requester is permitted to make such an output request. If this check is successfully completed, the data is released to the information storage program. This program is responsible for creating the information access keys and storing the data in the data base. Memory address verification and memory bounds register checks are also utilized by this program as are the controls for IOCP and the bulk file controller to ensure the integrity of the data being stored.

Terminal Input Processing

The user may enter any information he wishes from a terminal unit. It is his responsibility to provide control code names to all logical records which he enters. An individual terminal unit (in a work station) will be assigned to only one user at a time. No other user's request will be serviced through the same terminal unit until the first user has released the work station by removing his user key and the other user has inserted his key and satisfactorily identified himself at the work station. This is required to assure accountability. In Appendix V the multi-user work station is considered.

An input record from a terminal unit (for example, read card or read console buffer) is received in response to the user program making a programmed call on the I/O Processing service routine. The entry of information is checked in the following manner to assure that inputs are associated with the proper user. A memory area sufficient for the input record is established by the I/O Processing routine and its bounds are included in the input descriptor. Also inserted in this descriptor is the user key pattern expected from the work station containing that terminal unit. When the descriptor is fully prepared it is transmitted to the IOCP which executes the input to completion and returns an input-complete descriptor (including the key pattern and unit identification) with an interrupt. The ECP, in response to the interrupt, checks that the input came from the expected unit and that the key pattern matches the one in the user's control profile. The Converter security routine develops the control code of the input logical record from the input control code name. The Check Control Code security routine assures that the control code is allowed for output to that user who entered the information. If this check fails, the entered information cannot leave the memory (either to the bulk file or as output to the user's work station) and the system has no use for it. If all the above checks are passed, the input area is released to the user program by making its area available to the user PRT.

In the commonly used input technique of assigning two buffer areas to be used alternately for input loading and input processing, completion of input processing from one area by a user program returns that area for more input while information in the other area is processed. The control code check is only required for the first physical record of a logical record containing multiple physical records.

Terminal Output Processing

Output to a terminal unit includes two checks: the user's control profile contains the control code of the data to ascertain that the request is permissible, and the security level for the terminal unit includes the security part of the control code of the data to ascertain the equipment qualifications for handling the data. The output program passes the descriptor of the output request to the IOCP for executing the output.

Requests requiring prolonged processing can be left in process; however, any output to a work station is withheld until the intended user is actually at an output station and has properly identified himself. If the output is too bulky for conveniently storing in memory, the appropriate I/O service routine will create a special file on disc or magnetic tape with the same security protection by control code as any other file in the system.

Overlay Control

When the ECP determines that a user's assigned memory area(s) must be overlaid for another user, or when a user program is finished using an assigned area, the Memory Overwrite Check security routine of the ECP Terminate routine will blank out the area and all registers, table locations, etc., associated with it. A positive check on the blank-out procedure will

be successfully completed before reassigning the area(s). This software overwrite method is not an efficient use of processor time and is recommended only for systems that are input/output limited. An alternative overlay method using a special descriptor was discussed in the IOCP description on page 50.

SAFEGUARDS ON THE ECP

Placing security protection trust in the ECP and its service programs implies that they are vulnerable to alteration. Thus, safeguards are required on penetration of the ECP. ECP loading has been described earlier in this section; alteration by system programmers and self-protection are discussed in the following subsection.

ECP Alteration by System Programming

The system programmer may prepare and debug a program which will eventually be used as part of the ECP or as a service program. This program is executed in user mode and has a control code name indicating its eventual purpose. Such a program will probably include privileged instructions and attempted execution of a privileged instruction will interrupt the program. The ECP will interpret the interrupt, identify in that user's control profile that he is permitted to prepare a program having potential ECP or service routine use, and provide interpretative execution of that privileged instruction using only unclassified information in response. In order to use this unclassified information, it may be necessary for the interpreter to substitute names or addresses from an unclassified record for those used in the program. The system programmer, to minimize his uncertainty as a result of substitution, should supply suitable unclassified material himself. If the program performance depends upon control code recognition other than unclassified, one can be selected from a simulated

classified control code set. One unclassified record for each such simulated control code can be included in the data base. These simulated control codes can be chosen to have all distinctive security levels, and they can be included in each system programmer's user's control profile.

Upon the system programmer's satisfaction that his new program is ready for addition as a part of the ECP or as a new service program, administrative permission must be obtained from a system supervisor for inserting this program. In order to include this program, the supervisor opens a physical lock in the bulk file hardware. Within this unlocked compartment is the switch to release the write lock on the area of the bulk file in which these control programs are kept. Also in this area is the switch to set (at program write time) the execute-only flag bit on each word denoting control program.

Thus, the supervisor decides and enters any new control program only after having been satisfied by the preparing system programmer that it does function as intended. To execute the new program, the write lock must be restored, the compartment locked, and the program read into high-speed memory with flag bit now properly set for execution as a control program.

ECP Self-protection

The ECP and its service programs are operated by using the set of tables and directories (described in Section IV) which contain pertinent configuration and work demand information for the user programs to be run. In addition, a PRT is created for each user. The required tables are established by an ECP service program which is executable in control mode only. When outside control mode, the integrity of the tables is preserved since only one PRT for the current user is within the PRT memory bounds. All other

tables and the ECP and service programs, as well as all memory areas assigned to other user's programs, are excluded from the current user by memory bounds registers. All memory areas in these tables privately allocated to a user for data (or programs in some cases) are verified to be unique to the user. This verification is accomplished by an indicator in the memory directory available only to the ECP. Thus, the ECP cannot be violated through indirect addressing via these tables.

CONTROL OF USER'S CONTROL PROFILES

Several service programs (the last five entries in Table 3) associated with control of the user's control profiles have not been discussed in this single-thread analysis of the operating system. These programs would fit into the analysis as a special category of user jobs. The only qualification on these user's programs is that they are service programs, executable in control mode only, and labeled with control codes available only in the supervisory user's control profiles. These programs perform the following functions on or for the user's control profile:

1. create or change user's control profile,
2. create or change user's authentication data in the user's control profile,
3. output user current security authorization (control profile) to the supervisor, and
4. deactivate user's control profile.

Detailed descriptions of these tasks are contained in Appendix I.

SUMMARY OF SOFTWARE RECOMMENDED FOR SECURITY CONTROL APPLICATION

The ECP and associated service routines are considered as a representative part of the control of any multiprogramming, multiprocessing EDP system. The security augmentations indicated in the "thin-thread" analysis are the only parts properly charged as added costs for security control.

Table 4 contains a listing of the ECP routines which have been specifically added to the ECP to perform security functions. Column 1 contains the name of the routines, column 2 is an estimate of the number of instructions required for this routine, column 3 contains an estimate of the number of instructions which will be executed each time the routine is activated, and column 4 contains an estimate of the frequency with which the routines will be executed. The summation at the bottom of the table is for comparison purposes and indicates that the security routines represent a small addition to the ECP. Of the total, less than 200 instructions are required as part of the ECP resident in high-speed memory.

Table 5 is a list of the tables specifically added to the ECP table requirements for security purposes. Column 1 contains the name of the table, column 2 is a description of the table contents and lists the assumed sizes for a representative system, column 3 shows the estimated average table entry size, estimated table entry totals are in column 4, and column 5 contains the total size requirements for the table. It should be emphasized that these assumptions are not based upon any actual system. The total words indicated in high-speed memory is 8500 where the control code name table is retained in memory. For smaller systems, this could be reduced significantly by judicious selection of the most frequently used control codes and their names for storage in high-speed memory, delegating the less used ones to a backup table in bulk storage which would be called in as needed.

Table 4. ECP Security Routines

<u>Security Routines</u>	<u>Size Instructions</u>	<u>Instructions Executed/Call</u>	<u>Used Each</u>
ECP Loader	10	80,000	System Startup
Check Memory Bounds	50	35	Memory area allocation
Memory Overwrite	20	5n+10	Memory release, n words
Select User's Control Profile	20	30	User's control profile call
Check Control Code	20	40	Twice/data buffer input, once/other I/O
Converter from Control Code	30	200	Terminal output
Converter to Control Code	30	200	Terminal input
Insert Security Fields	20	15	Input/output setup
Check User at Input Complete	10	10	Input complete interrupt
Create or Change User's Control Profile	100	1,000	As required by supervisor
Create or Change User's Authentication Data	50	50	As required by supervisor
User Current Security Authorization	80	1,000	As required by supervisor
Activate User Control Profile	100	100	Work station startup
Deactivate User Control Profile	25	70	Work station shutdown
Total	535 instructions		
Estimated ECP Size in High-Speed Memory	8,000 instructions		
Estimated ECP plus Service Programs	35,000 instructions		

Table 5. Tables Added to ECP for Security Purposes

<u>Other Memory Space for Security</u>	<u>Assumed in Representative System</u>	<u>Average Size Words</u>	<u>Assumed Number</u>	<u>Total Words</u>
Control code name table	0.75 word control code + 3.75 word average security and need-to-know English control code name, plus 0.5 word	5	1000	5000
User control profile	100 control codes plus 20 words for authentication information and user key pattern	120		
	Total users (in data base)		50	6000
	On-line users (in high-speed memory)		16	2000
System user table	1 word for key pattern plus 1 word for control	2	50	100
Control code on all records, both data block and program segment	1 word header addition per record of average length 250 words (in data base) (in high-speed memory)	1		
			10^5	10^5
			10^3	10^3
Security field of IOCP descriptor	100 terminal units, space for 2 descriptors each; extra fields include control code, key pattern, redundant bounds	2	200	400

Taken together, the software additions represent a modest increase in the control overhead required for program execution and in the size of high-speed memory desirable to minimize the increase in response time to an on-line user's program because of security checks.

SECTION VI

IOCP APPLICATION FOR SECURITY CONTROL

In order to demonstrate control of input/output operations through the input/output control processor (IOCP), four distinct information flows having different security checking requirements exist. Both input and output are considered with terminal units and bulk files, via the bulk file controllers. The IOCP receives descriptor commands prepared by a Processor in the control mode, while executing an ECP or an I/O service routine. Descriptors provide control for security checks, high-speed memory addressing and bounds checks, and peripheral unit connection and selection. Returned descriptors indicate to the ECP the unit status at termination of an I/O action by either normal completion of record transfer or error condition. Record transfers are checked within the IOCP for maximum security. The security control afforded by the IOCP is, in large measure, redundant to that provided by the ECP and service routines. The IOCP provides assurance against single hardware errors since the security checks by the processor are repeated in the IOCP.

INPUT FLOW THROUGH IOCP FROM BULK FILE

When a program requires a record input from the bulk file, a series of checks is performed on the record by the ECP, the IOCP, and the bulk file controller, before the record is released to the program in memory. With Figure 7 as a guide, the following is a block-by-block description of the checks performed on an input record coming from the bulk file.

Figure 7. Input Flow Through IOCP From Bulk File

Beginning at (1) in Figure 7, the ECP information retrieval routine provides and sets up the size, control code, and file address of the desired record. This information is processed (2) by checking the control code from the retrieval information with the requestor's user control profile. If the control code is not found, an illegal request alarm (A2) is activated and access is denied. If the control code is found, an ECP routine is called (3) to assign an input area for the record and prepare the IOCP descriptor with the bounds for the input memory area, the file address, size, and control code for the desired record. This descriptor is passed to the IOCP, where it is stored in the unit control word of the I/O memory (4). The IOCP selects the buffer for the bulk file controller and passes the address of the record (5). The bulk file controller verifies the address (6) and initializes the sequence control by holding the address in the sequence register and reads the data of the record (7).

The data and the verification of the unit identity are accepted by the IOCP (8). The first part of the data contains the control code of the record. This control code is compared (9) with the control code from the previously stored descriptor. If they differ, an error interrupt (9A) is generated and an illegal access alarm (9B) is activated, which terminates the operation before any user access to the data is granted. If the control codes are the same, the verified unit (10) identity is compared with the identity of the unit contained in the previously stored descriptor.

If the units differ, the action is the same as (9A) and (9B) described above. If the units are the same, the memory address (11) is formed from the descriptor and a memory bounds check (12) is performed. If the access is out-of-bounds, action is the same as (9A) and (9B) described above. If the address is in bounds, the address is passed to memory (13), followed by store data (14). The IOCP checks the size field in the descriptor to see if the input is complete (15). If so, an input complete interrupt (24) is generated and the ECP services this interrupt by checking for the presence of an end signal (25) and that the control code in the first word of the record is present in the requesting user's control profile. If either is improper or absent an illegal access alarm (26A) is generated and access is denied. If both checks pass, the record is released (27) to the user's program by placing its area in the user's program reference table entry and making it available. This completes the input operation.

If the input is not complete (15), the memory address and the record size field are updated (16) for the next access. The IOCP (17) selects the bulk file controller for the next access. The bulk file controller (18) selects the desired file unit, connects to the desired file and verifies the connection, and reads the next data of the record (19). If this data does not contain a sequence control word (20), it is passed to IOCP (23) as described below.

If this data contains a sequence control word (20), the bulk file controller checks the control word (21) to see that it is the one expected. If not, a mislinked interrupt (21A) is generated by the IOCP and an error alarm (21B) is activated by the ECP, which desires further access. If the sequence indicator is proper, an end physical record signal (22) is generated for checking (25) that the end of the physical record is the same as memory requirements call for. The data with the verification of the unit identity is passed to the IOCP (23) where the unit identity is verified (10) as described above. At this point, action continues from block (10) in a programmed loop, until successfully reaching block (27), as described above, or until encountering an error exit which terminates the operation.

OUTPUT FLOW THROUGH IOCP TO BULK FILE

When a program requests to output a record to the bulk file, a series of checks is performed on the record and data flow by the ECP, the IOCP, and the bulk file Controller, before the record is put on the bulk file. Using Figure 8 as a guide, the following is a block-by-block description of the action performed in the output of a record going to the bulk file.

Beginning at (1), an ECP routine checks the control code of the record with the user's control profile. If the control code is not found in the user's control profile, the output is not allowed and an error alarm action is signaled. If the control code is found in the user's control profile, the ECP routine (2) is called to prepare the IOCP descriptor with the bounds of the output memory area, the file address, size field, and control code of the record. This description is passed to the IOCP, where it is stored in the unit control memory (3). When the bulk file controller is free, it is selected by the IOCP (4) and the unit and file identification is passed to it. Blocks 5 and 6 represent the bulk file controller verifying the bulk file

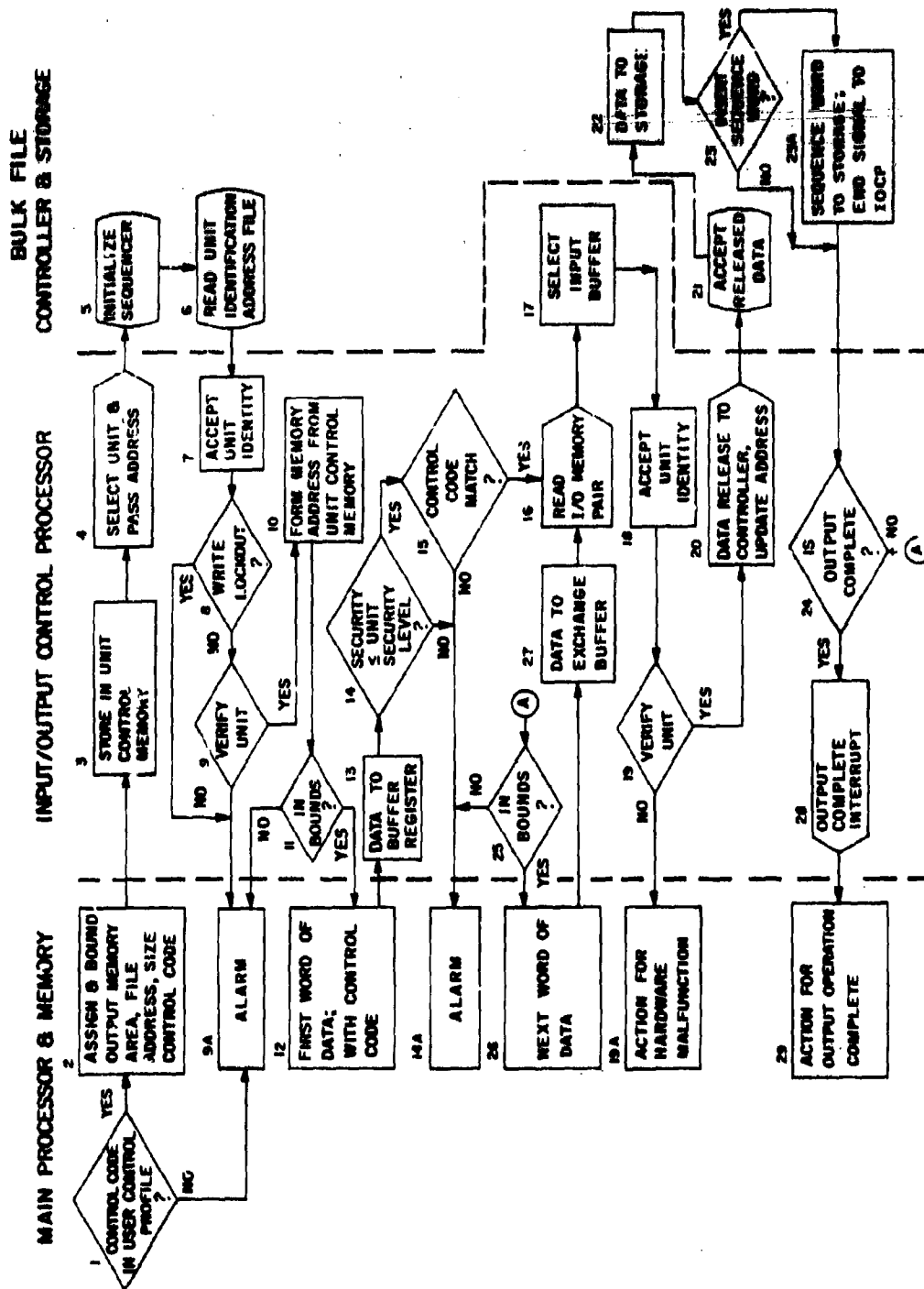


Figure 8. Output Flow Through IOCP To Bulk File

unit identification, loading the sequence register with the file starting address, and passing the unit address identification (and write lock-out signal when appropriate) back to the IOCP (7). If there is a write lock-out signal (8), an error alarm (9A) will be activated. If not, the unit verification signal (9) is compared with the original unit requested. If they are not the same, an error alarm (9A) is activated. If they are the same, the memory access address (10) is formed from the previously stored descriptor and a memory bounds check (11) is performed. If the access is out of bounds, an error alarm (9A) is activated. If not, the first data access is made (including control code (12)) from the high-speed memory, and stored in the IOCP buffer register (13). Block 14 checks that the security part of the control code of the data accessed is no higher than the maximum security classification allowed for the terminal equipment. If the equipment is not permitted to process this data classification, an error alarm (14A) is activated and access is denied. If the equipment is permitted to handle this data classification, the control code (15) of the data accessed is checked with the control code from the descriptor previously stored in the I/O memory for the unit. If they are not the same, an error alarm (14A) is activated and access is denied. If they are the same, the unit is read from the I/O memory (16) and the input unit buffer paired with the output unit buffer (17) is selected. The identity for the requested unit is checked with the identity of the unit in the original descriptor (18 and 19). If the units are not the same, an error alarm is activated for hardware malfunction (19A). If they are the same, the current data area is released to the bulk file controller and the address for the next memory access is updated (20).

The data is stored in the bulk file controller's buffer (21). The bulk file controller passes the data to the bulk file (22) after verifying its address, and then the bulk file controller checks to see if a sequence word should be inserted (23). If so, a sequence word is sent to the bulk file (23A), and an end of physical record signal is sent to the IOCP. If this is not the time for a sequence word, control is passed directly to the IOCP.

If the output is complete (24), an output complete interrupt (28) is generated which initiates ECP action (29) for servicing the interrupt. If there is more output, the

next address requested is checked to assure that it is still in bounds. If out-of-bounds, an error alarm (14A) is activated. If in-bounds, the next data is accessed in memory (26) and stored in the IOCP data buffer register (27). At this point, action continues from block (18) in a programmed loop, until successfully reaching block (29), as described above, or until encountering an error exit which terminates the operation.

INPUT FLOW THROUGH IOCP FROM TERMINAL UNIT

When a user inputs a physical record from a terminal unit, a series of checks is performed on both the user and the data by the ECP and the IOCP before the data is released for the user program. Guided by Figure 9, the following describes the action performed on the input coming from a terminal unit through the IOCP to memory, commencing with the user first identifying himself to the ECP as desiring permission at a work station.

Beginning at (1), the user inserts his identification key in the key pattern generator terminal unit, activating the work station. This action creates an interrupt (2) from the work station to the IOCP which enables the reading of the key pattern (4) from the work station into the IOCP and stores the key pattern in the IOCP unit control word of the I/O memory (5). The IOCP then generates an interrupt (6) to main memory (the ECP) which requests the IOCP to identify the unit and user (7), causing the original interrupt. The IOCP responds by passing the key pattern and unit address (8) to the ECP (9) which stores the information and interprets the key pattern to find the corresponding user's control profile. If the user's control profile (10) is not open to the system (original activation of the work station), an authentication process (10A), probably involving a special I/O, is activated to authenticate the user (10B). Successful completion opens the user's control profile (10D) to the system and returns control to (2) as described above. If the user fails to respond properly to the authentication routine (10B), an error alarm (10C) is activated and access to the system is denied.

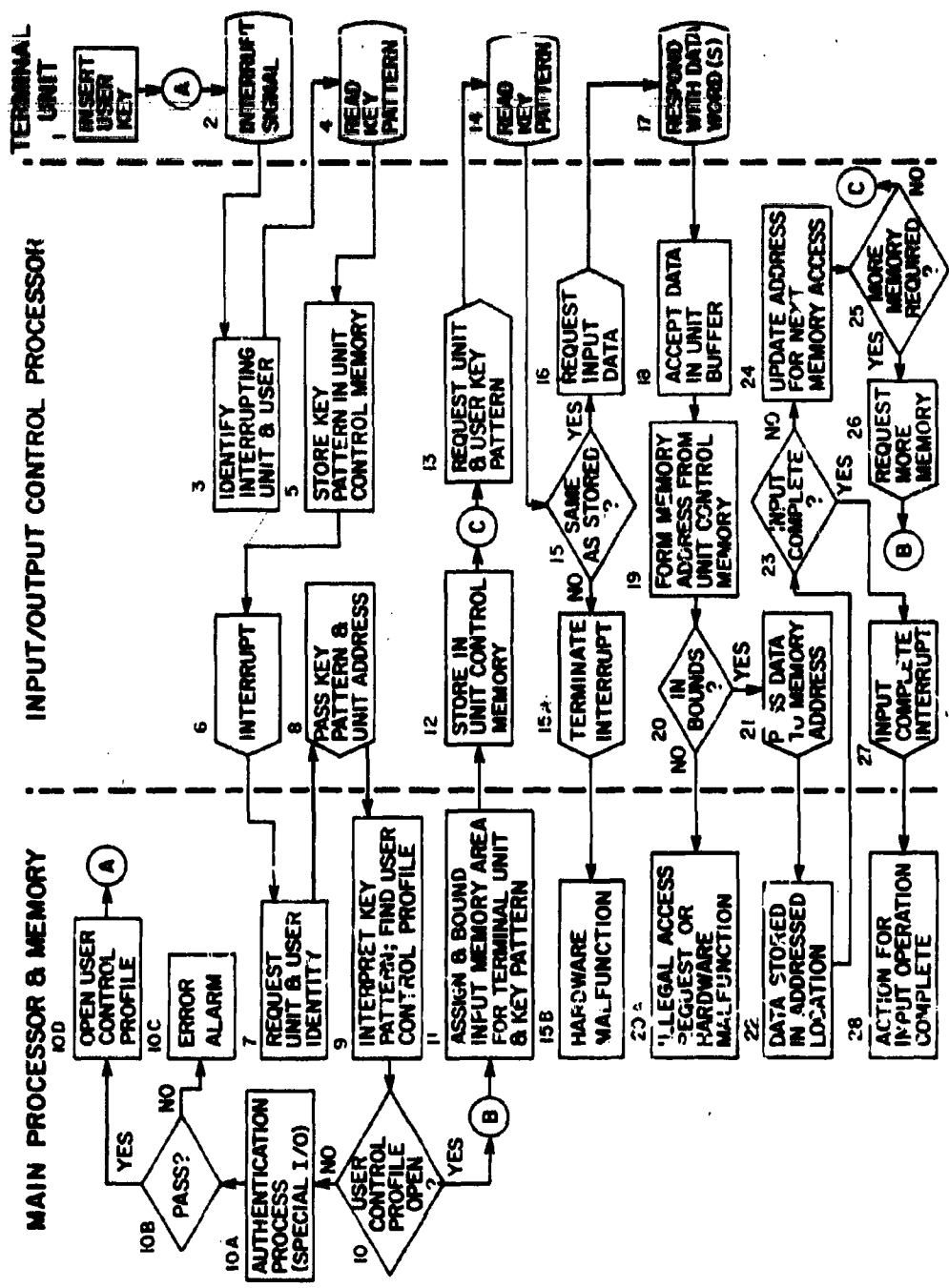


Figure 9. Input Flow Through IOCP From Terminal Unit

If the user's control profile (10) is open to the system, the ECP continues processing by allocating a memory area (11) for the input and setting up a descriptor containing the address and size of the assigned memory area. The previously stored unit identification and user's key pattern is passed to the IOCP where it is stored in the unit control memory (12). The IOCP again requests the user's key pattern (13) from the unit. After key pattern return from the selected unit (14), it is compared with the previously stored unit and user's key pattern (15). Any difference results in an interrupt (15A), which is interpreted by the ECP to be a hardware malfunction (15B) and the process is terminated. If user's key pattern for the unit is the same as previously stored, the IOCP requests the unit to send the input data (16) which it does (17).

The data is first stored in the IOCP unit input buffer (18). It is transferred to the data word for that unit in the I/O memory. When this data word is full, the associated control word furnishes the memory address (19) from the previously stored descriptor, and verifies that it is within the memory bounds (20). If not, the ECP interprets an IOCP interrupt for either illegal access request or hardware malfunction (20A) and the processing is terminated. If the address is in-bounds (20) the data is stored in memory (22). When this input record is complete (23), an input complete interrupt (27) is generated for an ECP program to take appropriate action (28). Otherwise, the IOCP updates the memory address (24) and the size field for the next access. A check is now made (25) to see if the allocated memory area has been completely used up. If it has, the IOCP requests (26) the ECP to allocate more area and control proceeds from block (11) as previously described. If there is sufficient memory, control is passed to block (13) as previously described. At this point, action continues in a programmed loop, until input is complete (28), as described above or until encountering an error exit which terminates the operation.

OUTPUT FLOW THROUGH IOCP TO TERMINAL UNIT

When a program requests to output a record to a terminal unit, a series of checks is performed on the record and data flow by the ECP and the IOCP, before the record is released to the terminal unit. Using Figure 10 as a guide, the following is a block-by-block description of the actions performed on the output of a record going to a terminal unit.

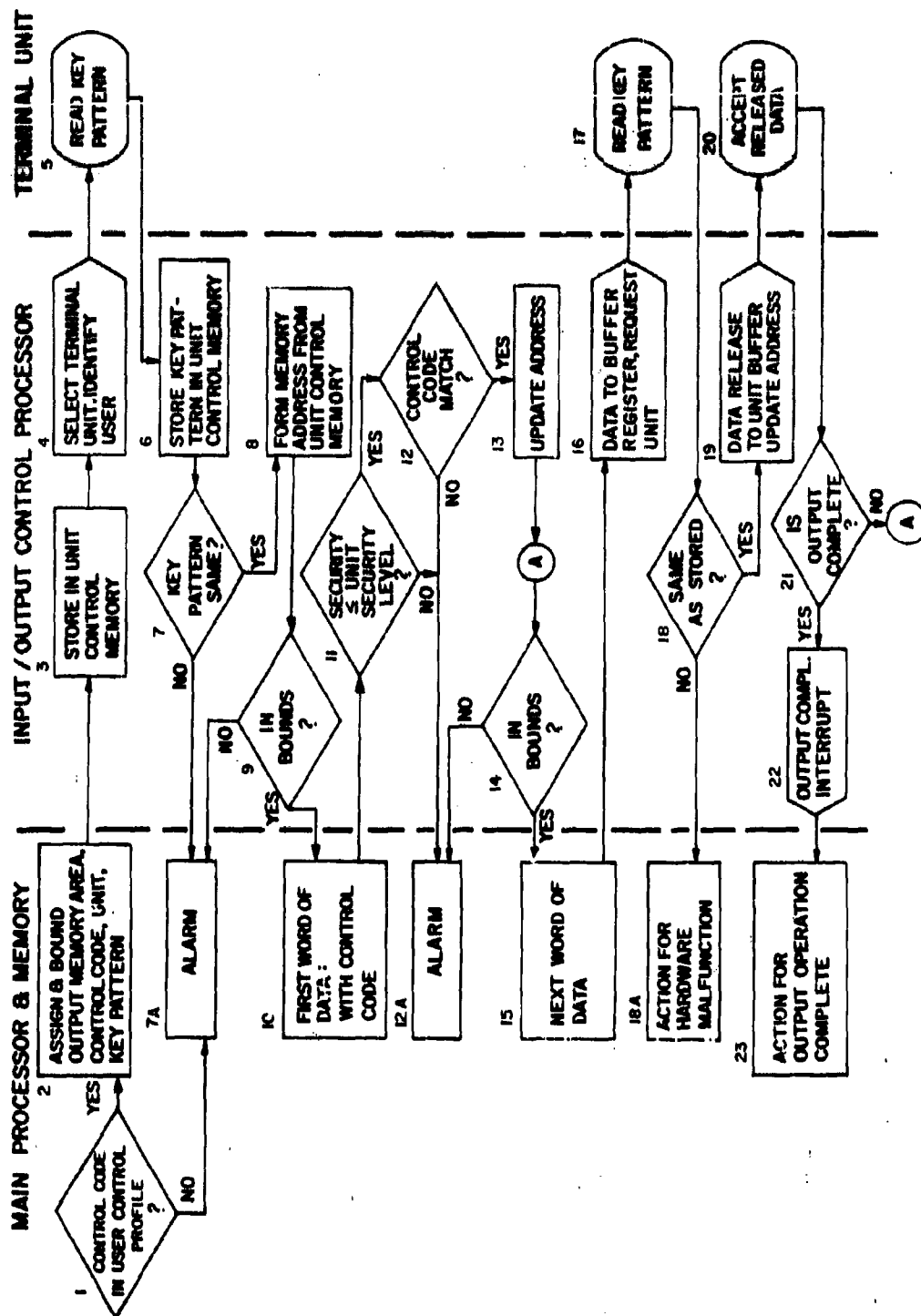


Figure 10. Output Flow Through IOCP To Terminal Unit

Beginning at (1), an ECP program compares the control code of the record with the user's control profile. If the control code is not found in the user's control profile, an illegal access error (7A) is activated and the output request is denied. If the corresponding control code is found, an ECP program (2) sets up the output descriptor for the IOCP containing the memory address, size field, control code for the record, and the unit address and key pattern for the output device. This descriptor is passed to the IOCP where it is stored (3) and used to select the key pattern generator terminal unit (4) for the destination work station in order to identify the user.

The selected terminal unit sends its key pattern output (5) to the IOCP where it is stored in that unit's data word in the I/O unit control memory (6). When all is present, it is compared (7) with the previously stored descriptor information. If they differ, an error interrupt (7A) is generated for the ECP and further access is denied. If the unit and user are the same, the memory address (8) is formed in the control part of the buffer register from the descriptor for that unit, and the memory bounds (9) are checked. If the address attempt is out-of-bounds, an error interrupt (7A) is generated for the ECP and further access is denied. If the access address is in-bounds, the first data word (10), which includes the control code, of the following record, is passed to the IOCP data part of the buffer register unit.

The maximum security level allowed for the output unit (11) is checked with the control code of the record to see if the terminal device is cleared to handle this security classification of information. If it is not, an error interrupt (12A) is generated for the ECP and further access denied. If it is cleared, the control code of the record (12) is compared with the previously stored control code in the descriptor. If they are not the same, action is taken similar to (12A) as described above. If the control codes are the same, since control codes are not externally used, the unit data buffer is cleared, the memory access address and size fields are updated, (13) and the address is bounds checked (14). If this access is out-of-bounds, action is taken similar to (12A) as described above. If the access is in bounds, the next data (15) is transferred from memory to the IOCP data buffer register (16), and the identity of the unit (17) is read and compared with the unit identity (18) previously stored in the descriptor. If they are not the same, an interrupt is generated

for ECP to take appropriate action for hardware malfunction (18A) and the operation is terminated. If this is the same unit (10), the addresses and size for the next memory access are updated and the data in the IOCP buffer register is released to the terminal device (20).

When output is complete (21), an interrupt (22) is generated for the ECP to take necessary action for output operation complete (23). If the output is not complete, control is passed back to block (14) as described above. At this point, action continues in a programmed loop, until successfully reading block (23) or until encountering an error condition that terminates the operation.

SECTION VII

SECURITY TECHNIQUE RETROFIT APPLICATION

In order to determine the feasibility for retrofit of the recommended security techniques to an existing data processing system, while at the same time minimizing cost and the possibility of revelation of other proprietary interests to Burroughs, the Air Force technical representative agreed that the Burroughs D825 be used as representative of an advanced but current system.

The D825 is an existing multiprogramming-multiprocessing system which is in use by all three military services. The Air Force application is the 416M BUIC system. It has many of the features recommended in this report.

The principal modules of the data processing system are the computer module, the memory module, and the input/output control module (IOCM). A memory exchange, distributed among the computer modules and IOCM's, is provided for communication with the memory modules. At least one computer module and one IOCM are required. The combined total of computer modules and IOCM's is limited to five modules, 16 or fewer memory modules may be used.

COMPUTER MODULE

Both user (normal) and control mode are available, indicated by the state of a single flip-flop. Setting for control mode is achieved only by an interrupt

occurring while in user mode. Control mode entry initiates the executive program (the automatic operating and scheduling program). Interrupt response stores job condition information to permit resumption of the user's job, performs a pre-determined sequence of operations to service the interrupt, and restores user mode for some user program execution after the interrupt service is completed.

The interrupt system is implemented with an interrupt register and interrupt masks. Twelve types of interrupts are standard, two are spare. The highest priority interrupts are automatic, bypassing the interrupt register; they are primary power failure and real time clock incrementing. Ten interrupts use the interrupt register. Four of these may be masked to prevent recognition by a particular computer. External requests from terminal devices are also recognized through the interrupt register. Each computer has 16 external request lines, over which communications needs are indicated from terminal units. An accepted interrupt initiates control relative to the interrupt address register which points to the base address of the table of starting locations of the interrupt response routines. Interrupt response is always initiated by reference to the entry in this table corresponding to the highest priority interrupt condition unserved.

Flag bits are not used in this system; their retrofit is a major system change involving either increased memory stack size and major timing and control changes in all modules, or reduced memory word size (say 7 instead of 8 characters) using the 8th for the flag bits with comparable major arithmetic and control changes primarily in the computer module. Replacing their function for control of execution or alteration of the control program could be partially achieved by coupling access to certain memory modules to being in control mode. This is not totally satisfactory since the service programs must be loaded on an overlay basis and thus, the IOCM must be able to write into these dedicated memory modules.

Privileged instructions, only executable in control mode, include loading the memory bounds register, the interrupt mask register, and the interrupt address register; interrupting another computer; returning from control to normal mode; halting a computer; and directing input/output command descriptors to the IOCM.

or output to a peripheral unit. It transfers data between a memory module (via the memory exchange) and one selected peripheral unit (via the input/output exchange). This operation takes place concurrently with, and independent of, a computer operation. The connections into the exchanges use switching interlocks contained within the IOCM itself. The control function of the interlocks is passive in the absence of conflicts and is fail-safe -- it is bypassed if the IOCM is disabled.

To provide memory bounds verification, an additional register, comparator, and necessary controls must be added to the IOCM. An additional descriptor is required to set the bounds, since there are not sufficient unused bits in the present one-word descriptors. Alternatively, two words could be accessed per descriptor. An 8-bit address precision on the bounds would require 14 cards. The addition of single word precision would require 24 cards.

To implement the security compare for inputs, each such device would be treated as two-way. The input would be compared character-by-character with an "Output" from memory provided that a comparator has been added to the IOCP. Key pattern comparison of the control content of one memory word against the control code head of a physical record is less readily achieved. It requires a buffer register to hold the entire control code, and a character-by-character comparator, plus an additional descriptor. The additional hardware would be 17 cards.

The addition to the IOCM to provide a 2-bit maximum security level check for each of the 64 possible connected terminal units could be implemented by a wired matrix with terminal unit selection on one coordinate and the unit maximum security level available on the output of the other. This output is compared against an added security indicator in the expanded descriptor. Fourteen additional cards would be required to implement this check.

These hardware modifications represent major changes to the IOCM which would probably require complete redesign, since the proposed additions exceed the amount of physical space available.

One pair of memory bounds registers is provided per processor which is used as read-only and guards against attempts to write into the included block while in user (normal) mode. Each register of the pair is 8 bits, allowing partitioning of the 64K memory with precision of 256 words.

In order to add the other two types of memory bounds registers with the same precision, 16 cards* are required. Since only one address at a time is developed, sharing the single pair of comparators already present among the three pairs of memory bounds registers is assumed.

In order to allow one word precision bounds registers with 14 bits each, 34 additional cards are required.

In order to make the mode flip-flop fail-safe, one additional card is required.

There is ample space for these alterations within the computer module.

MEMORY MODULE

Each memory module contains 4096 words of 49 bits each, including parity. Parity is stored and read, not checked. When a word is read, parity is checked by the receiving computer module or IOCM. Addition of parity checking applied to each 12-bit syllable (address or data) used for external parallel communication requires five additional cards. The parity comparison would be checked on input receipt as well as memory read.

INPUT/OUTPUT CONTROL MODULE

The IOCM is a limited purpose processor. It is driven by instructions contained in descriptors prepared by a control mode program executed in any computer module and received via a memory module. The IOCM controls any type of input

*A card contains approximately the same components as 4 equivalent flip-flops.

Two IOCM's are included on separate racks in one physical cabinet. They share the switching interlock circuitry which is placed on one of the racks, thus, leaving thirty-six additional card spaces in the more crowded rack.

SWITCHING EXCHANGES

The switching exchanges are organized about the receiving module; thus, no information intended for other modules is ever electrically received by an ineligible module. Consequently, a module may be placed in an off-line status and released for maintenance to personnel with a lower security clearance than the security classification of the information being processed in the adjacent on-line modules. A group of modules may even be placed off-line and operated together as a separate unit.

The physical grouping of cabinets consists of a single line with cabling routed through the sides to adjacent cabinets. Coaxial cabling between modules in non-adjacent cabinets is run through the intervening cabinets, which are used merely as cable troughs; no electrical connections are included. This design minimizes cable lengths and electromagnetic radiation.

There is normally no maintenance requirement for observation of signals on any cable other than for those actually used by the module, even though these are physically accessible. It would be possible to provide a separate cable trough between non-adjacent modules to eliminate this opportunity. Unless the cabinets are physically segregated it appears that administrative control is required over maintenance personnel access to cabinets; therefore, external cable ducts are not recommended.

The software techniques for system security control recommended in this report are directly adaptable to this modular system. Its automatic operating and scheduling program (AOSP) has been used as a pattern for the basic features suggested for the ECP. In this system, dynamic memory allocation is done by the AOSP for a job by allocating objects as requested. Data and program areas are separated. Each area has a base address register.

Other objects necessary but not currently within these areas are readied by macro calls on the AOSP. Multiple users may share common programs operating with private data areas. Programmed lock-outs of access to data objects are achieved by software. A program may request the AOSP to lock-out all other users from a data object. The inclusion of additional hardware memory bounds registers would provide positive control of this capability.

The AOSP coordinates and controls all input and output operations. Programs make macro requests on the AOSP for I/O operations. Control type I/O macros exist in which a program can request exclusive use of a terminal unit or file by being granted control of its status.

The AOSP provides fail-safe recovery control of detected errors on malfunctions, using the interrupt system as initiator.

SECTION VIII

EVALUATION OF ALL SECURITY TECHNIQUES CONSIDERED DURING THE PROGRAM

In the course of this study program, many possible techniques applicable to data processing systems were considered for security control. A system of hardware and software self-checking techniques was selected as offering the best assurance of fail-safe data processing of classified information within the constraints of present and future data processing system development. At the same time these techniques permit sufficient user flexibility so that he can perform his job effectively and efficiently. For the future, modular, multiprogramming, multiprocessing systems can include many of the currently recommended software techniques as part of the hardware design. Such techniques as memory bounds registers, locking of flag bit setting access, and bulk file content checking will both reduce system operation costs and provide a data processing complex best suited to the growing needs of the intelligence community.

Not all of the techniques studied during the program, however, were felt to be necessary for inclusion in the recommended system. Tables 6 and 7, therefore, summarize the hardware and software techniques studied, their essential applications, and their advantages or shortcomings. A page reference appears in the table with each of the techniques which were thought important enough for discussion within the body of the report.

Table 6. Hardware Techniques Considered for Security Protection

<u>Technique</u>	<u>Application</u>	<u>Protects Against</u>	<u>Comment</u>
Alarm associated with privileged instructions	Processor without control mode	Unauthorized use of instructions which defeat software control	Potential retrofit: application (151)
Bounds from program reference table	Relocatable user programs	Addressing into blocks outside user program	Special form of indirect addressing allows memory bounds register loading (41)
Cryptography	Magnetic tape or other portable storage	Mismounting and using	Not required in the physically secure: EDP system (160)
Dedicated high-speed memory	Physical separation of EDP, service, and security programs	Alteration by user	Not generally recommended; decreases memory allocation flexibility (159)
Encryption-decryption	Remote work stations	Communications penetration (garbles misroutings)	Only for communications security (26)
Execute-only flag bit set	Bulk file memory blocks	Unauthorized establishment of ECP service, or security programs	Installed in physically locked compartment (54)
Flag bits	Memory words	Control of memory bounds setting, identify program type, security label	Execute-only memory bounds load (36)
Interrupts to ECP-fixed location	Calls on control programs	User bypassing security checks	Allows only one place for beginning of control (35)

Table 6. Hardware Techniques Considered for Security Protection (Cont'd)

<u>Technique</u>	<u>Application</u>	<u>Protects Against</u>	<u>Comment</u>
Jump-trap flag bit	Memory block delimiter	Exceeding memory block by single-word-at-a-time address incrementing	Overly constricts addressing (32)
Maximum security level check	I/O terminal units	Release of information with higher security level than physical security of hardware allows	Redundant limit check to software (49)
Memory block erase and compare	Memory block	Sensitive information remaining in memory block when reassigned to another user	Recommended if required so frequently that equivalent software result is unacceptable (50)
Memory bounds registers	Memory addresses, read/write, real only, or execute only	Unauthorized access by user program or I/OCP	Apply to all addresses developed in user programs (40)
Memory protection by user number comparison for each access to memory	Memory block compared to single user requiring access	Overwriting by program error or misaddressing; makes difficult shared reference among some but not all users	Less efficient than flag bit memory bounds for many users sharing memory
Output lockup	All output terminal units	Unauthorized output attempts	Not recommended; privileged instruction for I/O is recommended instead (159)
Page associative memory with page limit	Relocatable single level storage	Addressing beyond allocated page	Not considered fundamental; the program reference table provides more flexibility

Table 6. Hardware Techniques Considered for Security Protection (Cont'd)

<u>Technique</u>	<u>Application</u>	<u>Protects Against</u>	<u>Comment</u>
Parity	Data and address interchange	Hardware errors	For all information interchanges between modules (36, 50)
Privileged instructions	I/O commands, special register setting	Uncontrolled I/O, user program executing control-changing instructions	Small set of instructions reserved for control mode only (38)
Processor modes	Restricts user program	User program executing privileged instructions, altering control tables, or system directories	Provides hardware differentiated software control (33)
Read only memory; write disabled	ECP as stored bulk file	User alteration of ECP	Allows only one place for mode control and execute-only flag bit instead (54)
Redundant addressing	Terminal unit addresses of IOCP	Undetected address error resulting in misrouting	Grouped by work station (49)
Security label flag bit or bits	Memory word classified/unclassified indicator, or control mode	Classified release to unclassified terminal or user program	Inefficient use of memory (38)
Security verification	Logical record	Misaddressing or mislinking physical records of logical record	Confidence in access to complete logical record (49)
Unit connection identification	Bulk file controller	Accessing error or mislinking within physical record	Redundant to address (53)

Table 6. Hardware Techniques Considered for Security Protection (Cont'd)

<u>Technique</u>	<u>Application</u>	<u>Protects Against</u>	<u>Comment</u>
User's key pattern generator	Work station user identification	Unauthorized processing request or output to work station	Representative of hardware for user identification process, (51)
Write lock	Bulk file memory blocks	Unauthorized alteration of stored physical records	Installed in physically locked compartment (54)

Table 7. Software Techniques Considered for Security Protection

<u>Technique</u>	<u>Application</u>	<u>Protects Against</u>	<u>Comment</u>
Authentication procedure	Individual user at work station	Identification of one user as another user	Fundamental but beyond scope of this study (85)
Challenge insertion in linked program lists	Linked program lists; insertion method is simple since only linkages are altered	Unauthorized use	Particularly applicable to string or block structured languages (76)
Confidence test programs	Program execution results compared against expected results	Hardware error	Part of normal diagnostic program design (78)
Content identification	Logical record header and trailer	Erroneous retrieval of parts of two logical records	Linking and control code labeling for each physical record is preferable (91)
Content verification	Data having some redundancy in coding	Obvious discrepancies in data fields, such as value out-of-range	No primary role in security, except as applied to control codes (q. v.)
Control code containing security and need-to-know	Information entities	Release of entities to user unless control code is in user control profile	Fixed length encoding for machine processing convenience (62)

Table 7. Software Techniques Considered for Security Protection (Cont'd)

<u>Technique</u>	<u>Application</u>	<u>Protects Against</u>	<u>Comment</u>
Dynamic allocation of memory	All named data and program segments internally assigned to actual memory blocks	User knowing absolute addresses	Should be part of future EDP system (87)
Equivalence preserving transformations	Machine language user program	User knowing the detailed form of his program	Possible area for future investigation (64)
Hash totals (block parity formed by adding each word of a block as if it were a number)	Long data entries such as data or program segments	Wrong addressed segment transfer, partial transfers	Hardware for block parity is more appropriate for bulk storage devices
In-process job identifier	Job incarnation for some user	Mis-association of incarnation with user	Should be normal part of job accounting
I/O by ECP or service program in control mode using privileged instructions	All I/O to peripheral equipments	User bypassing control checks made by ECP prior to release of output data, or unauthorized release of an input buffer area to a user program	Fundamental to security control on input/output (89)
Item counts	Data or program segments, fixed or variable length	Wrong number of items in record or file	Often available for addressing and memory bounds; adds little
Physical record linking	Program segment or data physical record sequence linking in bulk file; control code in each	Mis-linking of successive physical records	Symbolic file naming and physical record linking; part of information storage and retrieval system design

Table 7. Software Techniques Considered for Security Protection (Cont'd)

<u>Technique</u>	<u>Application</u>	<u>Protects Against</u>	<u>Comment</u>
Procedure oriented languages for user communications	User queries on the data base	Illegal constructs, job interaction, machine language alterations of procedures	Too restrictive for fundamental technique; desirable as way to minimize man-machine language interface
Program compilation or assembly monitor	New program	Illegal constructions or data references; program debugging	Not sufficient; only an aid to control programming activity (65)
Randomly inserted program challenges	Private program for one user who alone knows the challenge responses	Unauthorized use by any other user	Suitable for any user program (70)
Program reference table	High-speed memory block assigned during allocation at job scheduling time	Unauthorized use of block; memory bounds associated in same word with actual block address	Protected from user alteration by PRT bounds register (67)
Redundant addresses	Program entry and sub-program linkages	Failure in some addressing hardware	Not recommended; memory bound registers serve comparable purpose with much more independence of hardware
Redundant programming	Control code - user control profile checks executed by different processors	Protects against some hardware errors, and partial program bypass	Multiprocessing can effectively use this without processing time penalty (65)
Security level	Equipments and inter-connection links	Classified information release through equipment not protected for the classification level	Hardware inclusion in IOCP, redundant software check to allow temporary reductions (95)

Table 7. Software Techniques Considered for Security Protection (Cont'd)

<u>Technique</u>	<u>Application</u>	<u>Protects Against</u>	<u>Comment</u>
Self-checking digit	Short data entries such as accession numbers, control codes, word counts	Manual entry or query by number mistakes; machine bit error	Not fundamental; primarily a source error check
Separation of program and data; programs placed in user-read only memory sections for processing	All compiled programs	Operation code modifying programs, program analysis and alteration without compiler-provided checks	Simplifies shared use of single copy of programs, and protection of programs from alteration
Sequence monitoring	Job execution by inserting progress indicators and checking their presence before continuation	Bypassing program control sections	Effectively accomplished by control mode processing, application to single-mode processors not fail-safe
Sort sequence	Ordered or numbered items or records	Manual entry, or query by number mistakes; machine bit error	Part of information storage and retrieval system design. No primary role in security protection
System Logs	All system use; user identification; file accesses; input/output to work station	Deters user exploration of system	After-the-fact detection of attempted security violations of suspicious patterns of use (70)
User's control profile	Permission for user actions and access to classified records	Improper action or access	Accessible only in control mode (63)
User job number	All inter-unit exchanges and processing for job incarnation for user	Internal job part misassociation; entire job being sent to other user	Necessary part of system logging (71)

SECTION IX
BIBLIOGRAPHY

1. Baran, P., "On Distributed Communications," Rand Corporation, RM-3765-PR, U.S. Air Force Contract No. AF 49(638)-700, AD444839, August 1964.
2. Barnett, N. L., and Fitzgerald, A. K., "Operating System for the 1410/7010," IBM, Datamation, May 1964.
3. Bobrow, D. G., and Raphael, B., "A Comparison of List-Processing Languages," MIT, Communications of the ACM, Vol. 7, No. 4, April 1964.
4. Bright, H., "Philco Multiprocessing System," AFIPS, Vol. 28, Part II, 1964.
5. Brooks, F. P., Jr., "The Future of Computer Architecture," IBM, AFIPS, Vol. 1, 1965.
6. Burroughs Corporation, "B5500 Information Processing System," Reference Manual, 1964.
7. Burroughs Corporation, "B8500 System Description," Report No. D-685, May 1965.
8. Burroughs Corporation, "Characteristics of the Disc File Master Control Program For the Burroughs B5500," Report No. 5500-21001, July 1964.
9. Burroughs Corporation, "D825 Automatic Operating and Scheduling Program," Report No. TR63-38B, August 1964.
10. Burroughs Corporation, "Operational Characteristics of the Processors for the Burroughs B5000," Report No. 5000-21005, Revision A, 1963.

11. Burroughs Corporation, "The Burroughs D825 Modular Data Processing System," Report No. TR61-58C, 1961.
12. Burrows, J. H., "Program Structure for Military Real-Time Systems," Mitre Corporation SR-122, U. S. Air Force ESD, Contract No. AF19(628)-2390, AD610318, January 1965.
13. Codd, E. F., "Multiprogramming," Advances in Computers, Vol. III, 1962.
14. Computer Command and Control Co., "Associative Memory Computer System - Description and Selected Naval Applications," CCC 25-101-11, U. S. Navy Contract No. Nonr 4068(00), AD466313, 10 April 1965.
15. Corbato, F. J., "System Requirements for Multiple Access, Time-Shared Computers," MIT, Project MAC TR-3, AD608501, May 1964.
16. Dennis, J. B., "Program Structure in a Multi-Access Computer," MIT, Project MAC TR-11, AD608500, May 1964.
17. Defense Intelligence Agency and Departments of the Army, Navy, and Air Force. "Security Controlling the Dissemination and Use of Intelligence and Intelligence Information Produced by Members of the Intelligence Community," DIAR 50-10, AR381-1, OP NAVINST 5500.39, AFR205-19, 28 September 1962.
18. Dion, F. A., "The TRW Two-Station, On-Line Scientific Computer," TRW/STL, RADC, TDR-64-392, U. S. Air Force, RADC, Contract No. AF30(602)-3097, AD609720, December 1964.
19. Fano, R. M., "The MAC System: The Computer Utility Approach," MIT, The IEEE Spectrum, January 1965.
20. Feldman, J. A., "Aspects of Associative Processing," MIT, Tech Note 1965-13, U. S. Air Force, Contract No. AF19(628)-500, AD614634, April 1965.
21. Ferranti, Lt'd., and Manchester U., "User's Specification on the Atlas Computer," CS 255, February 1960

22. Fuller, R. H., "Content-Addressable Memory Systems," UCLA, 63-25
U. S. Navy, Contract No. Nonr 233(52), AD417644, June 1963.
23. Galentine, Paul G., Col., USAF, "Advanced Programming Development:
A Survey," Air Force Systems Command, USAF and Computer Associates,
Wakefield, Mass., U. S. Air Force, Contract No. ESD-TR-65-171,
AD614704, February 1965.
24. Gallenson, L., and Weismann, C., "Time-Sharing Systems: Real
and Ideal," SDC, SP-1872, ARPA, Contract No. SD-97, AD612940,
March 1965.
25. Harlow, J., "Research in Information Retrieval," ITT, 5400-TR-0096
U. S. Army Electronics Laboratories, Contract No. DA-36-039-SC-90787,
AD461099, July 1964.
26. Holt, A. W., "Program Organization and Record Keeping for Dynamic
Storage Allocation," Applied Data Research, Inc., Communications
of the ACM, Volume 4, No. 10, pp. 422-431, October 1961.
27. IBM "Applied Research Program, Aerospace Intelligence Data System
(AIDS)," (final report with appendixes), U. S. Air Force, RADC,
Contract No. AF19(626)-10, AD619149, May 1964.
28. Informatics, Inc., "Executive Control Program (ECP-1A),"
U. S. Air Force, RADC, Contract No. AF30(602)-3045,
AD610817, January 1965.
29. Iram, D., "Error Control Through Coding" (8 volumes), U. S. Air
Force, RADC, TDR-64-149, Contract No. AF30(602)-2958,
AD609471, July 1964.
30. Kochen, M., "High-Speed Document Perusal," IBM, U. S. Air Force,
Contract No. AF49(638)-1062, AD285255, May 1962.
31. Korotkin, A., et. al., "Indexing Aids, Procedures, and Devices,"
GE, U. S. Air Force, Contract No. AF30(602)-3435, AD616382,
April 1965.
32. Lechner, R., et. al., "A Feasibility Study of Future Militarized
Digital Computer Requirements," Sylvania, U. S. Army Electronic
Labs., Contract No. DA-36-039-AMC-03276(E), AD480375,
December 1964.

33. MIT, "Project MAC," ARPA, Contract No. Nonr-4102(01), AD465088, July 1964
34. Mauceri, A. J., "Feasibility Study of Personnel Identification by Signature Verification," RADC, TR-65-33, "North American Aviation, SID65-24, U. S. Air Force, Contract No. AF30(602)-3493, AD617615, April 1965.
35. Mitchell, G. J. "Preliminary Description of a Memory Protection Feature for the CDC 1604," Institute for Defense Analyses, Princeton, Working Paper No. 95, IDA-CRD Log No. 8323, April 1965 (Revised).
36. Nisenoff, N., "An Optimizing Study of a Modular Digital Computer System Final Report," Vol. I, and Vol. II, Honeywell Corp., U. S. Army Contract No. DA-36-039-AMC-03275E, AD462042, April 1965.
37. Nolan, J. F., and Armenti, A. W., "An Experimental On-Line Data Storage and Retrieval System," MIT, TR-377, U. S. Air Force, AD615658, February 1965.
38. Oliphant, C., "Operating System for the B5000," Burroughs, Datamation, May 1964.
39. Proshan, F., "Optimum Redundancy Under Multiple Constraints," Boeing, D1-82-0253, AD408393, May 1963.
40. Saltzer, J. H., "Project MAC, CTSS Technical Notes," MIT, MAC-TR-16, ARPA, Contract No. Nonr-4102(01) AD: None, March 1965.
41. Shaw, J. C., "JOSS: Experience With an Experimental Computing Service for Users at Remote Typewriter Consoles," Rand Corporation, P-3149, Dept. of Defense, AD615843, May 1965.
42. Singer, T., Schupp, P., "Associative Memory Computers From the Programming Point of View," Mitre Corporation, W-5492, U. S. Air Force, Contract No. AF33(600)-39852, AD416301, August 1963.
43. Simmons, R. F., "Natural Language Processing and the Time-Shared Computer," SDC, SP-1974/001/00, ARPA, Contract No. SD-97, AD615531, April 1965.

44. U. S. Air Force, "Security Safeguarding Classification Information,"
AFR 205-ID, May 1965.
45. Zimmerle, D. F., "MILDATA Study, National Cash Register Co.,
No. 24-4Q, U. S. Army Electronic Labs., Contract No. DA-38-039
AMC-03277(E), AD457809, October 1964.

APPENDIX I

SECURITY ROUTINES

The software routines described here are those which are added specifically for security control to an electronic data processing system operating under the envisioned executive control program. Most control programs have security violation implications if they malfunction. However, their designs are not unique to security protection, but are required for operation of the system as described, and thus are not detailed here.

The following routine descriptions are machine-independent and assume only rudimentary instruction processing capability packed one instruction per word for the memory space estimates.

NAME: ECP LOADER

ABSTRACT: Check routine to ascertain the ECP was correctly loaded.

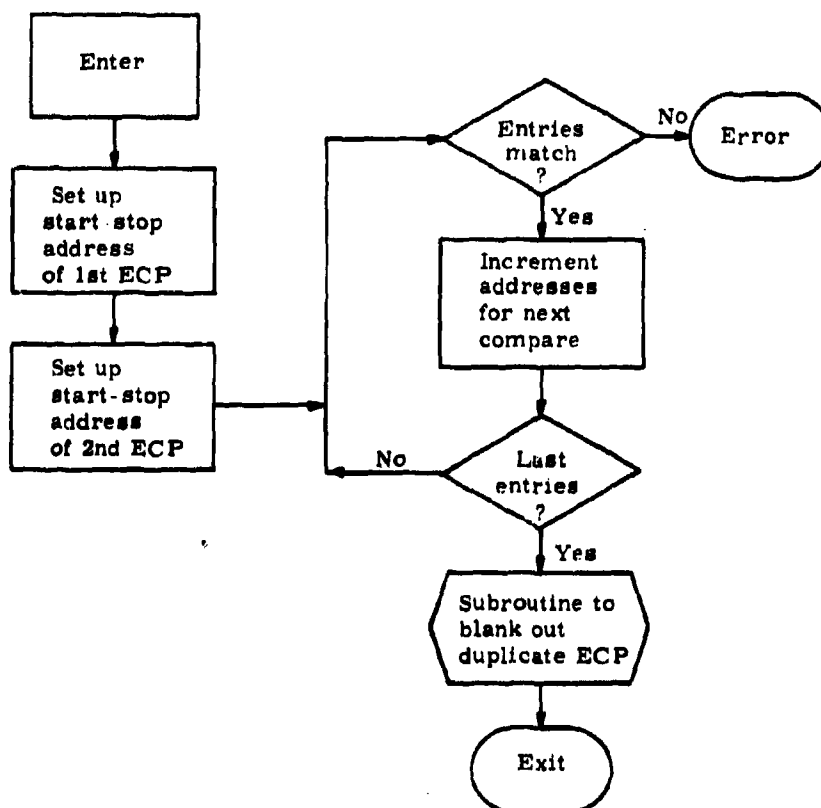
PROGRAM DESCRIPTION: This routine is an instruction by instruction compare of the ECP. It is automatically activated after the ECP has been loaded in duplicate by separate I/O channels, from separate disks into separate memory modules.

MEMORY REQUIREMENTS: 10 instruction loops plus temporarily duplicating ECP in memory (typically 8,000 words).

INPUTS: A duplicated ECP loaded into separate memory modules.

OUTPUTS: Either blank out the duplicate ECP and continue the ECP functions, or halt on error condition and reload the ECP.

LOGICAL FLOW DIAGRAM:



NAME: CHECK PROCESSOR MEMORY BOUNDS

ABSTRACT: To set and test the settings of memory bounds register.

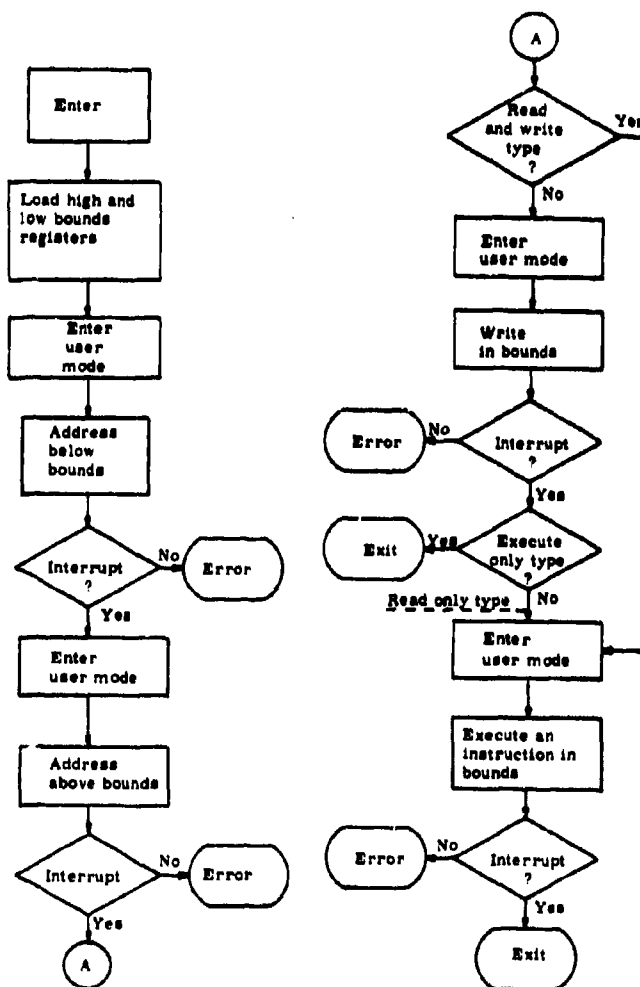
DESCRIPTION: This routine sets and then tests the setting of memory bounds registers by both out-of-bounds access attempts and in bounds but wrong access type yielding executed alarm interrupts.

MEMORY REQUIREMENTS: 50 instructions.

INPUT: Allocated memory block use type, low and high addresses.

OUTPUT: Memory bounds registers properly protect memory block, or error alarm.

LOGICAL FLOW DIAGRAM:



NAME: MEMORY OVERWRITE

ABSTRACT: To blank out memory and check to see that it was blanked out.

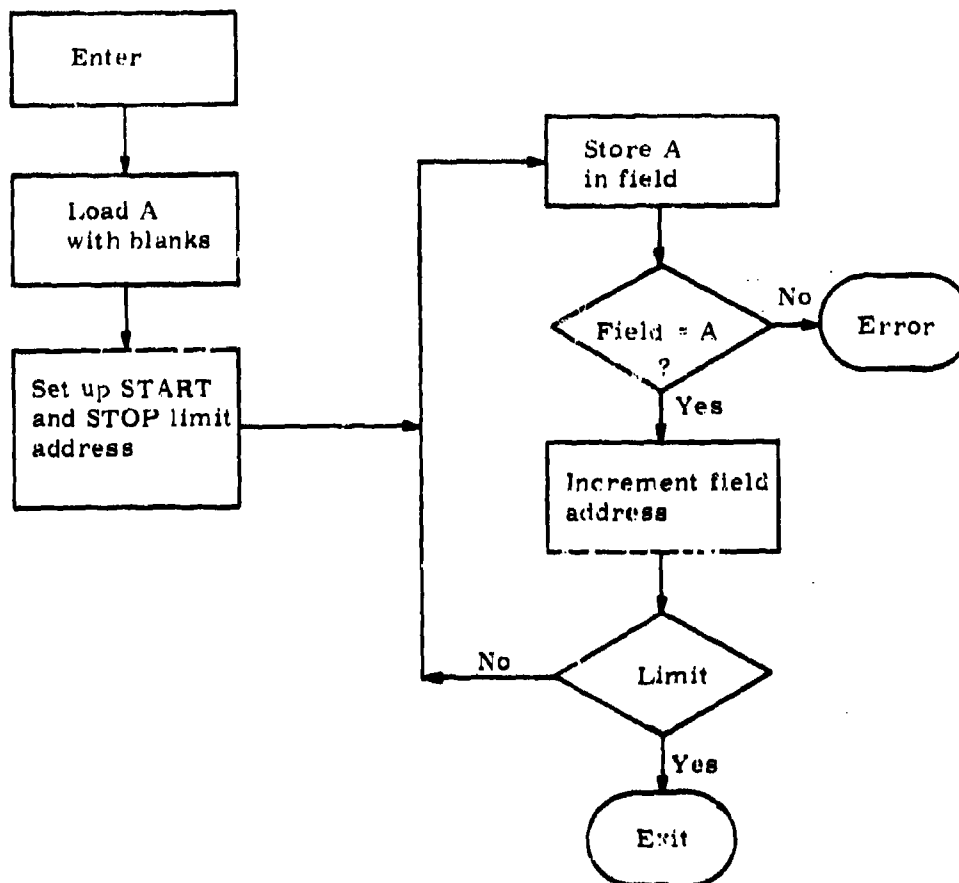
DESCRIPTION: When a user releases a memory area or after memory reallocation, this routine blanks out the memory area and checks to see that the area was blanked out. *

MEMORY REQUIREMENTS: 20 instructions.

INFUT: High and low addresses of the memory area to be blanked.

OUTPUT: Continue processing, or error.

LOGICAL FLOW DIAGRAM:



*The same thing can be accomplished by using the IOCP as described in another section of this report.

NAME: SELECT USER CONTROL PROFILE

ABSTRACT: To furnish the correct user control profile as required.

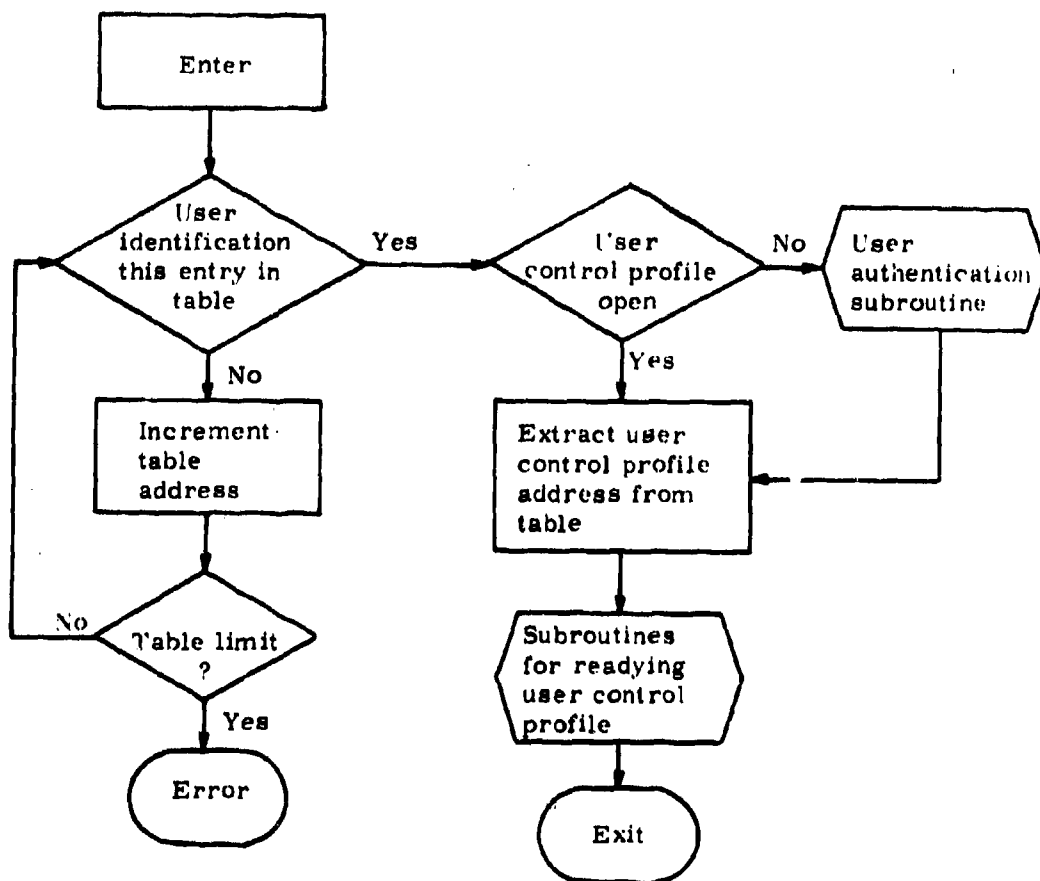
DESCRIPTION: This routine compares a user key pattern with entries in the system user table and, on finding a match, extracts the corresponding address for the user control profile and brings it into memory if not present.

MEMORY REQUIREMENTS: 20 instructions.

INPUT: User key pattern or identification, system user table, subroutines for user authentication and input as required.

OUTPUT: User control profile available in memory or error condition.

LOGICAL FLOW DIAGRAM:



NAME: CONTROL CODE CHECK

ABSTRACT: To verify that the control code of the record is in the user control profile for the type of action requested before giving user the requested access.

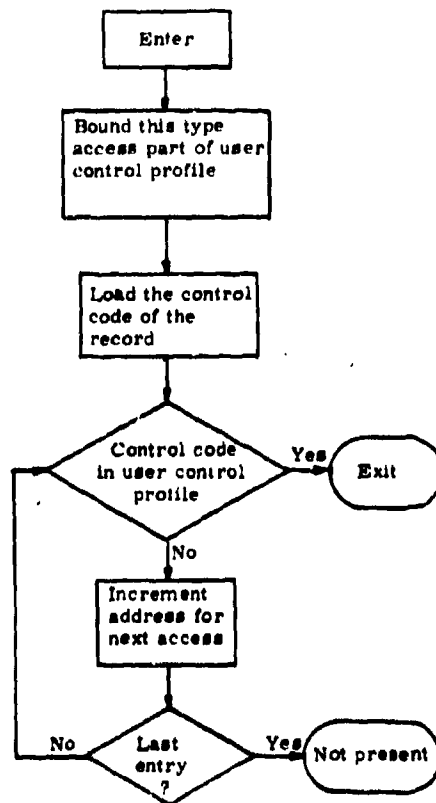
DESCRIPTION: Compare the control code of the record with the control code list for the access type in the user control profile. If there is a control code match the request is allowed. If not, the request is denied.

MEMORY REQUIREMENTS: 20 instructions.

INPUT: User control profile of calling user, control code and access requirements of the record.

OUTPUT: Either continue processing or error alarm if control code is not present.

LOGICAL FLOW DIAGRAM:



NAME: CONVERTER FROM CONTROL CODE NAME

ABSTRACT: To convert a control code name to the corresponding control code, or to assign new control code as appropriate.

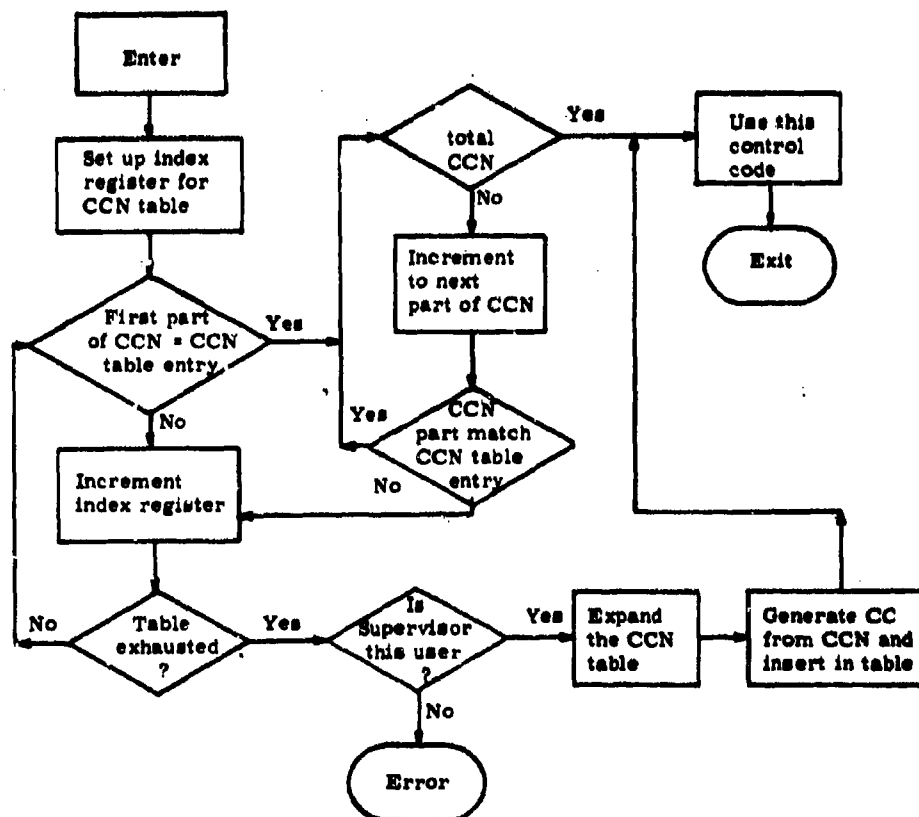
DESCRIPTION: This routine searches the control code name table for a match of the input control code name. If a match is found, the corresponding control code is substituted. If not, and the user is a supervisor, a new entry is established in the table and a new control code is assigned and used, otherwise an error condition exists.

MEMORY REQUIREMENTS: 50 instructions.

INPUT: Control code name and control code name table.

OUTPUT: Control code for the control code name and an updated control code name table if appropriate.

LOGICAL FLOW DIAGRAM:



NAME: DEACTIVATE USER CONTROL PROFILE

ABSTRACT: To indicate the user control profile is closed to the system.

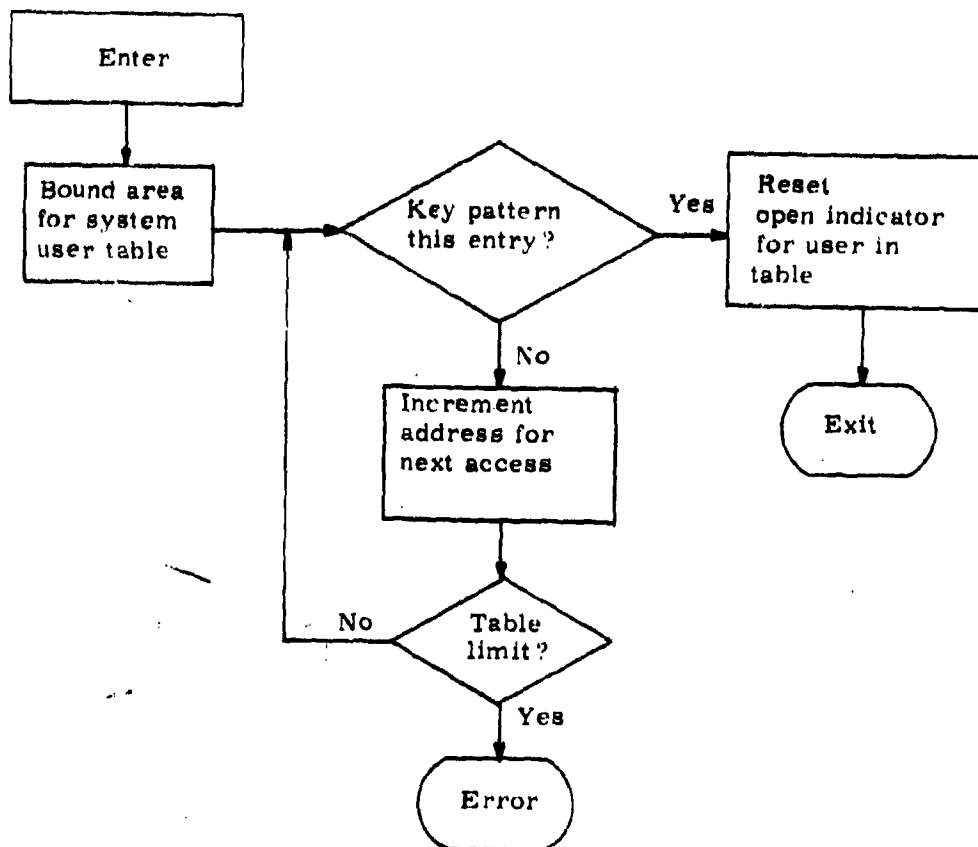
DESCRIPTION: After a user has closed his work station (removed his key) an interrupt signal is generated. This routine identifies the user last associated with that work station, locates the user in the system user tables and resets the open indicator to indicate that this user control profile is closed to the system.

MEMORY REQUIREMENTS: 25 instructions.

INPUTS: Interrupt descriptor containing user key pattern, and the system users table.

OUTPUTS: Closed entry in the system users table.

LOGICAL FLOW DIAGRAM:



NAME: ACTIVATE USER CONTROL PROFILE

ABSTRACT: To indicate the user control profile is open to the system.

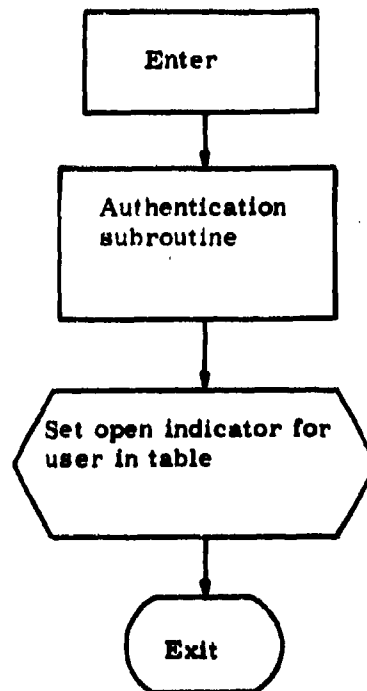
DESCRIPTION: After a user has been identified and authenticated to the system, the open indicator in the system user table is set to indicate that this user control profile is open to the system.

MEMORY REQUIREMENTS: 100 instructions plus a table containing one entry per user of the system.

INPUTS: User key pattern and the system users table.

OUTPUTS: Activated entry in the system users table.

LOGICAL FLOW DIAGRAM:



NAME: CONVERTER TO CONTROL CODE NAME

ABSTRACT: To convert the control code to the control code name.

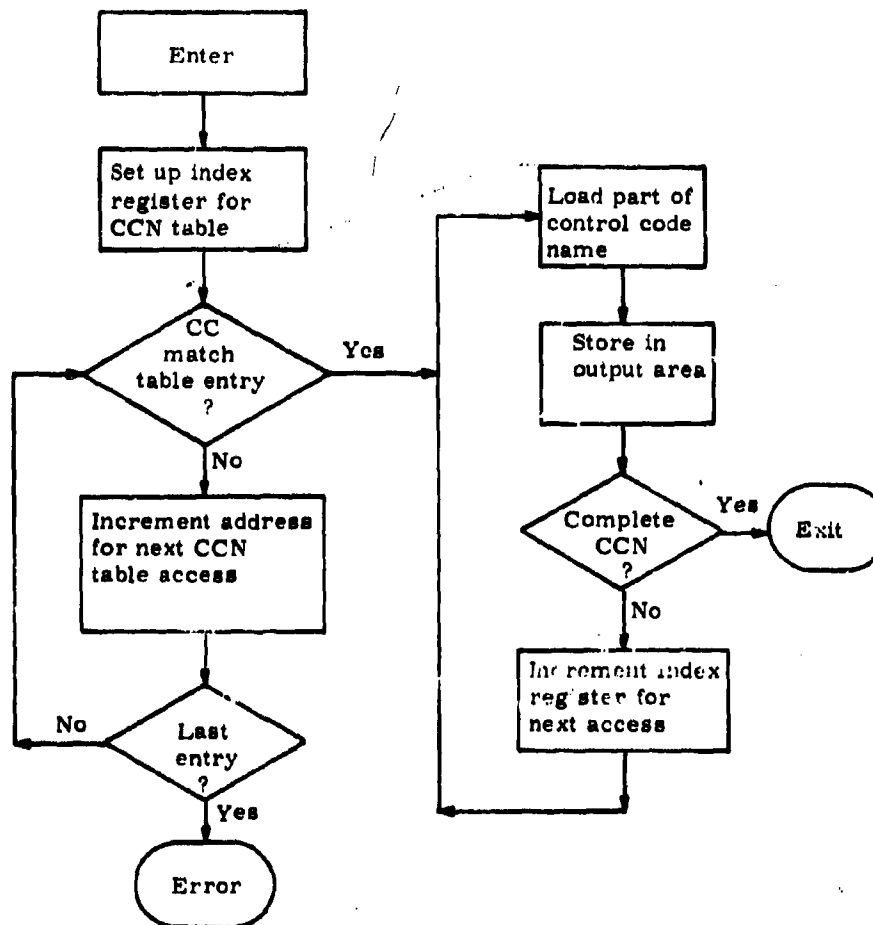
DESCRIPTION: This routine searches the control code part of the control code name table for a match of the input control code. If a match is found the corresponding control code name is substituted. If not, an error condition exists.

MEMORY REQUIREMENTS: 30 instructions.

INPUT: Control code, and control code name table.

OUTPUT: The control code name corresponding to the input control code.

LOGICAL FLOW DIAGRAM:



NAME: USER CURRENT SECURITY AUTHORIZATION

ABSTRACT: To output the user current security authorization to the supervisor responsible for verification.

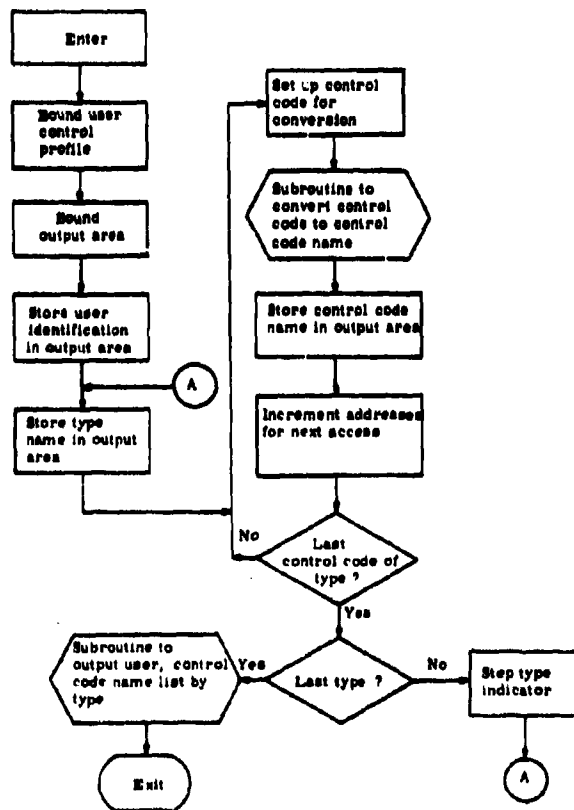
DESCRIPTION: This routine outputs a user identification and the control code names corresponding to control codes by use in a user control profile, to the responsible supervisor.

MEMORY REQUIREMENTS: 80 instructions.

INPUTS: User control profile, control code-control code name routine, output routine.

OUTPUTS: Control code names corresponding to a user authorization.

LOGICAL FLOW DIAGRAM:



NAME: CREATE OR CHANGE USER AUTHENTICATION DATA

ABSTRACT: Change users response required in the authentication procedure.

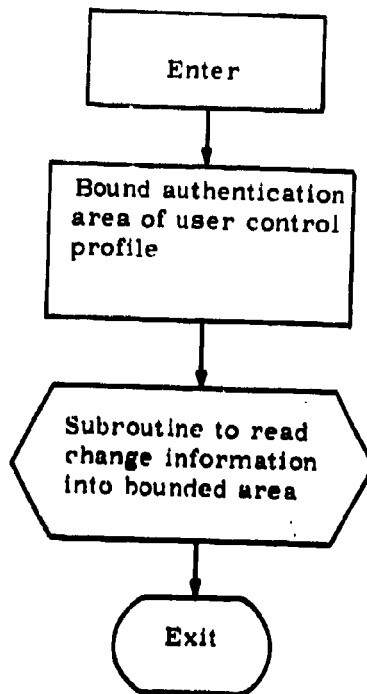
DESCRIPTION: This routine changes the user data entry response required in the authentication procedures by forming a table of the required responses in the selected user control profile.

MEMORY REQUIREMENTS: 50 instructions.

INPUTS: User control profile, input routine, and new authentication information.

OUTPUTS: User control profile containing updated authentication information.

LOGICAL FLOW DIAGRAM:



NAME: CREATE OR CHANGE USER CONTROL PROFILE

ABSTRACT: Set up authorized control codes in a user control profile.

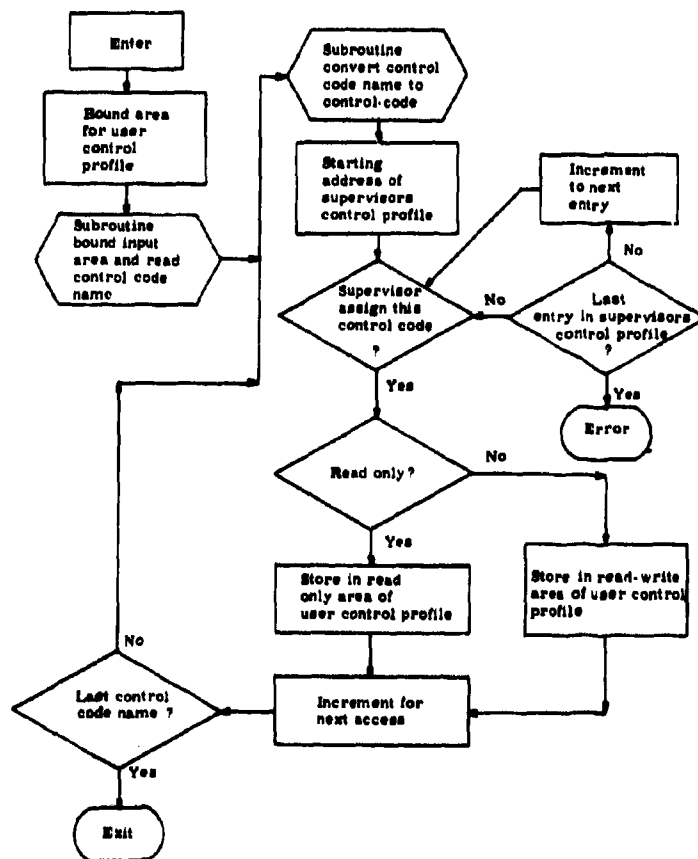
DESCRIPTION: A supervisor's routine for entering those control code names which he authorizes for a user and forming the user control profile control code lists.

MEMORY REQUIREMENTS: 100 instructions plus input area and area for user control profile.

INPUTS: User key pattern and authentication information, supervisor's user control profile, desired control code name list each with access type, converter and input subroutine.

OUTPUTS: User control profile.

LOGICAL FLOW DIAGRAM:



NAME: INSERT SECURITY FIELDS IN IOCP DESCRIPTOR

ABSTRACT: To set up I/O descriptors for the IOCP.

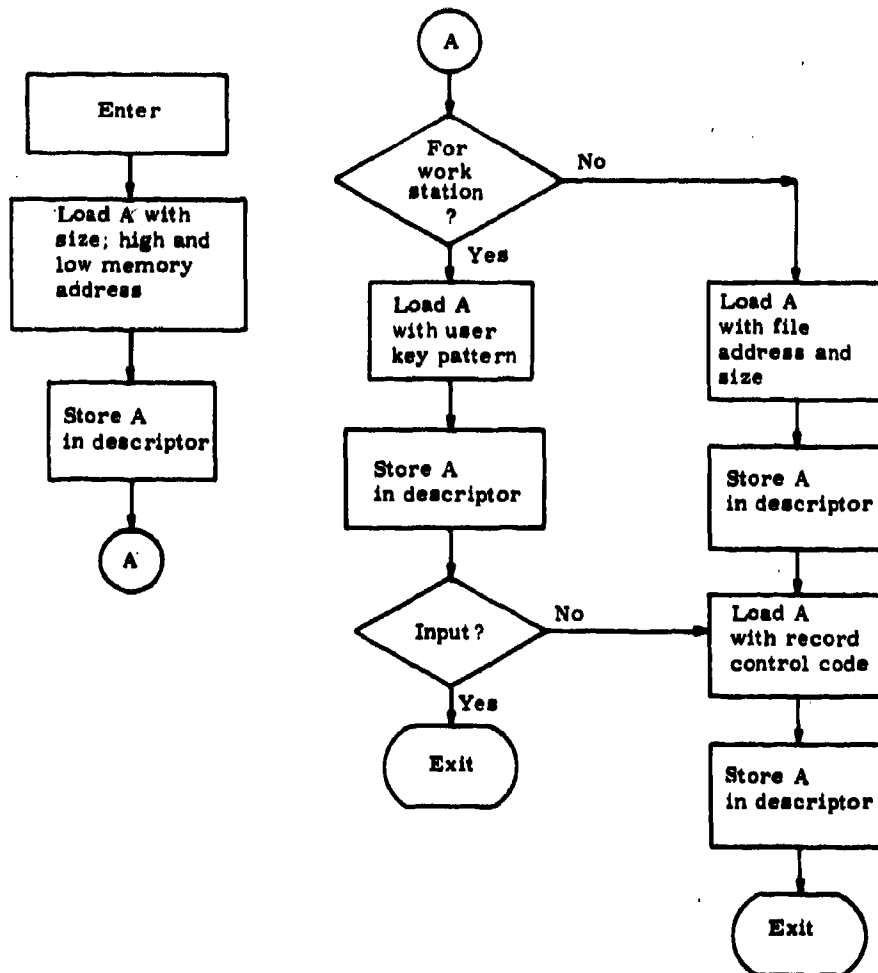
DESCRIPTION: This routine inserts the control code, memory bounds, and user key pattern in the I/O descriptor as required.

MEMORY REQUIREMENTS: 20 instructions.

INPUTS: Control code or user key pattern, redundant memory size description, unit, use, user, memory area, and file address.

OUTPUTS: Completed I/O descriptor.

LOGICAL FLOW DIAGRAM:



NAME: CHECK USER AT INPUT-COMPLETE

ABSTRACT: To verify the originating user is the same as the terminating user.

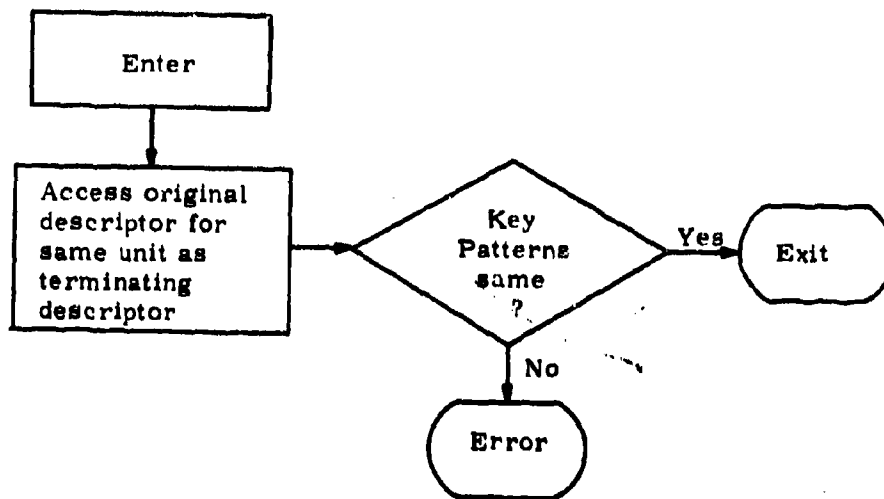
DESCRIPTION: This routine compares the unit and user key pattern furnished by the I/O complete descriptor with the unit and user key pattern stored from the original request. If they are the same, processing continues. If they are different, there is an error alarm.

MEMORY REQUIREMENTS: 10 instructions.

INPUTS: Originating and terminating descriptors containing user key patterns, and unit identification.

OUTPUTS: Either processing continues or error alarm.

LOGICAL FLOW DIAGRAM:



APPENDIX II

ALARM ASSOCIATED WITH PRIVILEGED INSTRUCTIONS

In a processor system without a control mode, the effect of the privileged instruction can be achieved by providing a security macro - a sequence of instructions preparing for execution of the privileged instruction. This macrosequence must be entered at its beginning, and processed in its entirety so that any necessary prerequisites for a privileged instruction will assuredly be fulfilled. If at any step in this macrosequence a deviation is sensed, an error alarm is given.

Let the set of instructions available in a processor be separated into four types:

N : Normal

P : Privileged

U : A unique instruction in its state context (may be type N)

T : Turn-off alarm.

Consider that the system normally operates with an alarm on. Figure 11 indicates the desired conditions in diagram form. Any user program can use normal instructions without consequence. A unique call for a security macro occurs by transfer (or indirect address) into a trap area of memory.

This entry point is the only one allowed for a security macro, as it initiates a unique predecessor - successor program sequence culminating in a turn-off alarm and privileged instruction execution.

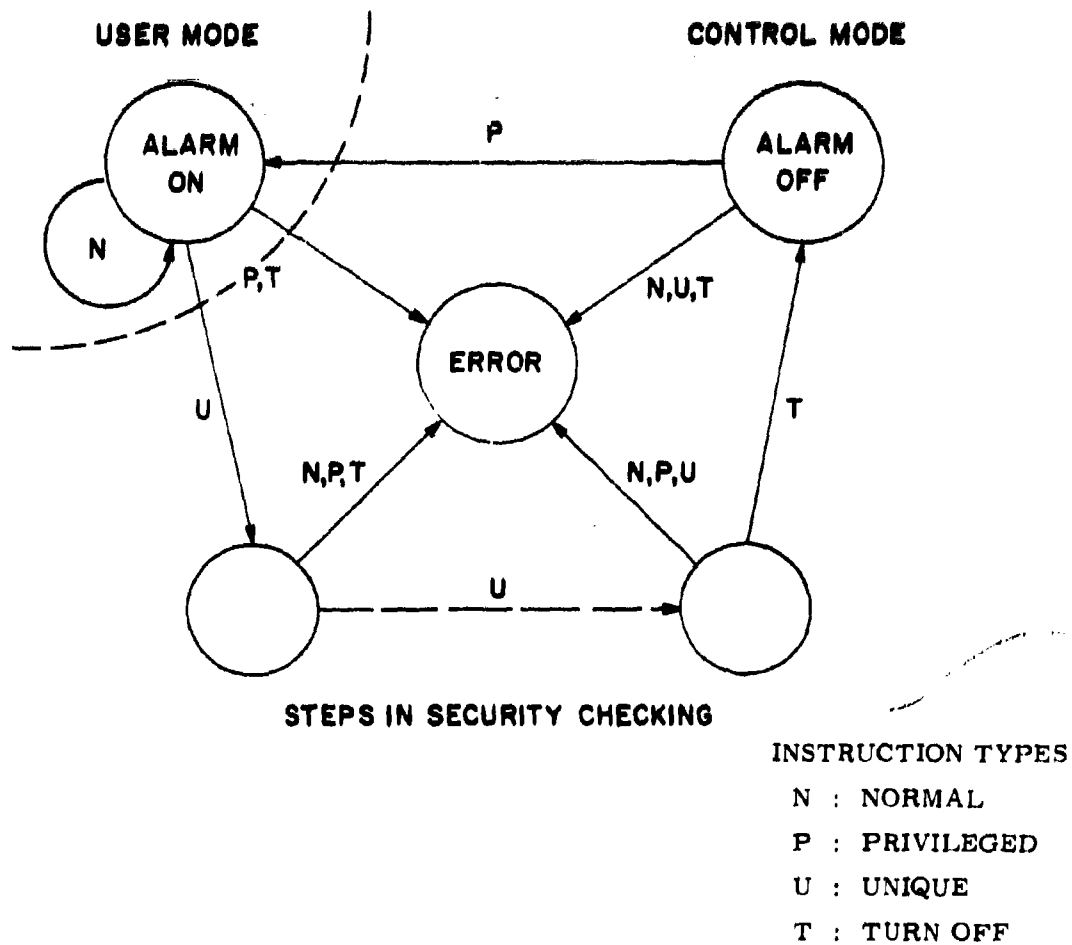


Figure 11. Fail-Safe Privileged Instruction Execution

Note that only one privileged instruction is executable with the alarm off, since its execution restores the alarm. Note also that any other instruction executed will give error. This is a form of fail-safe design using a combination of hardware and software. The error state is terminal in this diagram, since appropriate corrective action in response to the errors is not being described.

The logic associated with security macros for fail-safe privileged instruction execution can be summarized into the following performance conditions. Any one of the alternatives is sufficient when present.

1. Entering a security macro:
 - a. Addressing from a program sequence outside a security macro into the trap area of memory reserved as starting locations for security macros.
2. Stepping to the next instruction with alarm enabled:
 - a. Executing a normal instruction outside a security macro;
 - b. Addressing into the trap area to enter a security macro; or
 - c. Within a security macro, fetching for execution the unique instruction allowed by the macro at this point.
3. Disabling the alarm:
 - a. Executing a unique "turn-off alarm" instruction in a security macro.
4. Enabling the alarm:
 - a. Executing a privileged instruction.
5. Error alarm:
 - a. Fetching a privileged instruction for execution from anywhere but its proper place in a security macro (i. e., with the alarm on);
 - b. Fetching any instruction for execution out of order in a security macro;
 - c. Fetching the "turn-off alarm" instruction for execution outside a security macro; or
 - d. The alarm remaining disabled for more than one instruction.

Figure 12 shows a functional diagram of an independent security macro box which provides multiple security macro protection by add-on retrofit to a presently

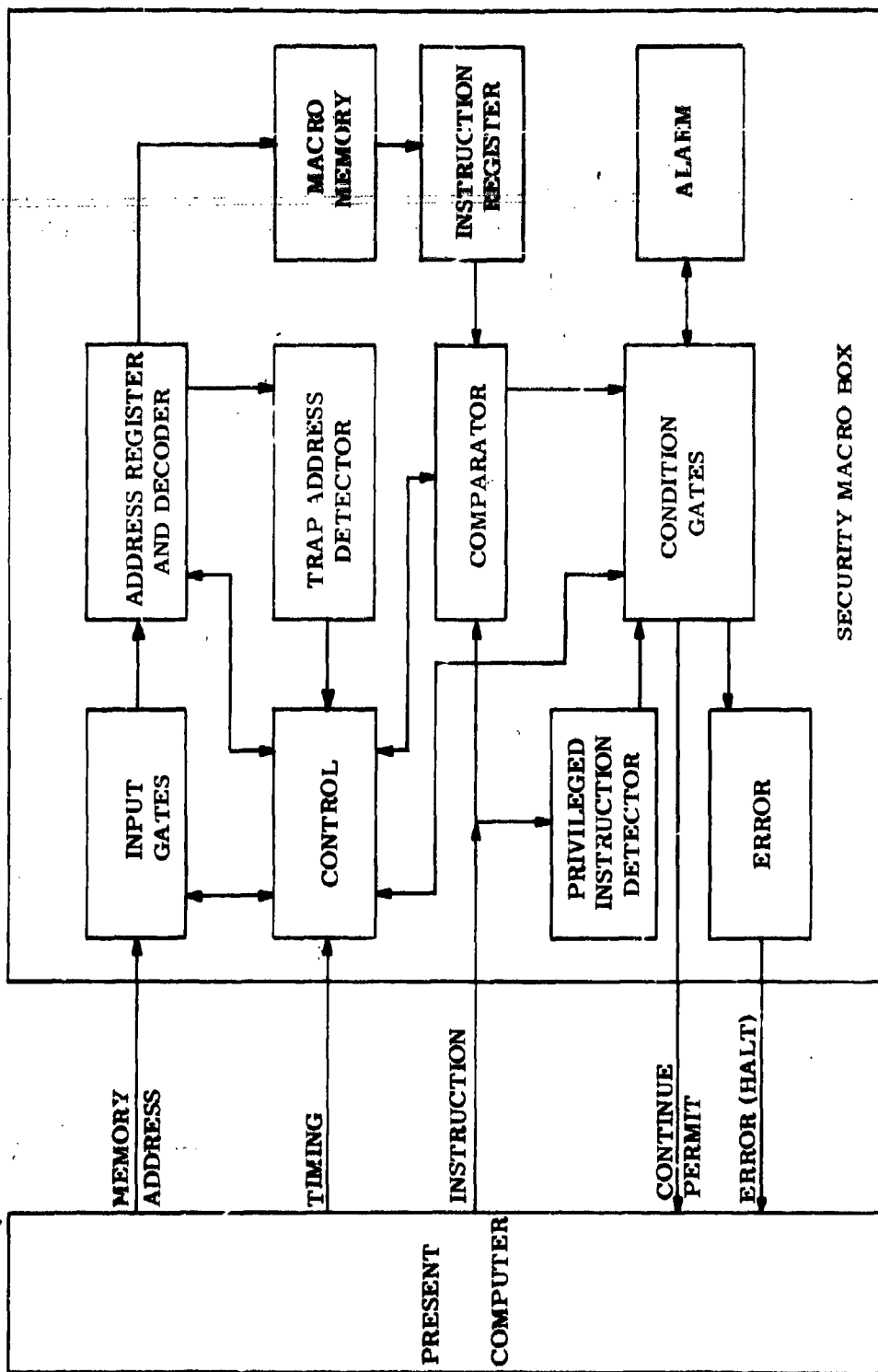


Figure 12. Functional Diagram of Security Macro Box

designed computer. This box includes a macro memory. It also includes input gates and an address register and decoder to select the proper instruction and necessary gates, drivers, and sense amplifiers to read an addressed word. The accessed instruction enters an instruction register from which a comparator matches it against the instruction from the present computer when in a security macro. The logic conditions previously described are used as gating to control the alarm, to determine when security macros are being processed, and to indicate when privileged instructions must occur. A privileged instruction detector is also required to give error indication at out-of-place occurrence. A trap address detector is used to initiate control and select a particular security macro.

Required signals from the present computer are the program address (from input gates to the address register or the register output itself); the fetched instruction (from input gates to the address register or the register output itself), and timing signals to indicate the presence of the above. The two principal signals may be time alternated in many computers, and may even pass through a common register, in either case, simplifying the compatibility problem. Ignoring any combination and assuming a 15-bit address, a 12-bit instruction code, and a 2-bit timing control, a total of 29 signals are sampled.

At least one (and preferably two) outputs of the security macro box to the present computer is required to indicate error (and desirably continue permission). The "turn-off alarm" instruction may be an "NOP" taken in context of the security macro if no unused order codes are available.

For the estimated system interfaces, the logical component estimate is 400 equivalent flip-flops plus a wired memory of 48 cores, sufficient for 1028 instructions in security macros. The addition of the present computer yields between 15 and 60 equivalent flip-flops plus interconnecting cabling, assuming power and space where needed are available.

An example of application of this technique is a user program (operating in the normal alarm-on state) call for file read. The trap resulting from this call changes the system state to the first instruction of the security verify macro.

The possible exits from this macro are: any failure to verify (an error requiring the ECP action), any unexpected instruction (an error action since only one is allowable per state of the macro), or the procedure completion at which time the alarm can be turned off, changing the system state. From the alarm-off state, the only allowable instruction must be a privileged one; the expected one establishes the file read. Any instruction from another group would be an error in keeping with the fail-safe design.

An extension to the security macro concept could provide levels of alarms. Depending upon the privileged instruction to be executed, the control could successively pass through more than one security verification sequence.

This concept of alarm states is closely related to multiple system modes (user and at least one control mode) but extends these modes to include the unique ordering of instructions in a security control macro, terminating with the execution of the privileged instructions.

APPENDIX III

MEMORY PROTECTION

Program and data protection in a multiprocessing EDP facility necessitates concern even in systems where all memory modules are dedicated to pre-specified uses. However, security warrants major concern in systems where the memory modules are dynamically allocated and partitioned among many users and the executive control program. Burroughs concept for memory protection is described in detail below.

GENERAL APPROACH

The general approach to the memory protection problem involves an independent, double checking procedure. The intent is such that each memory read or write operation will be individually validated. In the case of read, the first check is of the address before access is obtained, and the second, is on the address-content after access has been accomplished. In the case of write, the first check is before or in parallel with the address formation, but before the actual writing. The second check, where justified, is a read of the information just written. In general, reading checks are for security, while writing checks are to afford protection of the data.

The intent is to make the two parts of each check as independent of each other as possible; however, in any particular application, there will necessarily be some commonality. The exact balance (for example, of hardware to software) is largely

a function of the basic processor system and its related program structure, although it is intended to accomplish the address check by hardware and the contents check by either parity checking hardware or specific content-matching using software. The two checks are intended to use different criteria in that the first check is on a physical location or address, and the second, is on the content. The first check is accomplished by means of control circuit paths, and the second, by means of data circuit paths.

Application to a Generalized System

Given this generalized approach to memory protection, the application to the various memory types that an electronic data processing system might have can be considered. For this system application, it is convenient to partition the hardware into two areas: the processor complex, which includes the processor and high-speed memory modules; and the input/output complex, which includes the input/output control processor and peripheral tapes, discs, card readers and punches, operating and maintenance consoles, and user data input or output terminal units and displays.

Many processors contain private scratch-pad memories, functioning to provide a small-volume, fast-access store for the particular job (program) being executed. Protection of the scratch-pad memory is an exception to the generalized approach. In a fail-safe manner, it is cleared every time it is no longer required by the particular program being executed. A special "block clear" feature, which permits the entire scratch pad to be cleared simultaneously can readily be incorporated in most memory stack designs by addition of some extra high-current drive signal on existing signal lines.

Protection of the main high-speed, random-access (thin film or core) memory is best accomplished by application of the generalized approach. It is this high-speed, random-access category or level in the hierarchy of memory types that is the working store. It contains programs, tables, records, and blocks of data being worked on by the processors. In data storage areas at least, it is advisable to provide access control by memory bound registers. It is also desirable to clear-between-bounds at the conclusion of each program so that no residual classified information is available to the next user of that memory area. Classified programs, if they exist, should similarly be cleared between bounds.

The input/output control processor, since it serves to match the generally low-data rate of peripheral or terminal units with the high-speed memory, will generally contain a buffer memory similar in organization to the main memory. Through this buffer memory passes the addressing information for both high-speed memory and peripheral devices, as well as all of the data and records that go to or come from the peripheral devices necessary to the internal operation and maintenance of the data processing complex. The buffer memory should be protected by redundant addressing such that no single bit error in addressing results in another allowable address.

Discs and drums are used for storage of large quantities of data and provide moderate access times. They may be partially protected by the input/output control processor which services them as described above. In addition, they have their own controllers which provide physical record address recognition for the bulk storage as well as data transfer control.

To dedicate each memory module or device to a particular program or analyst would be uneconomical and tend to defeat the advantages of centralized multiprocessing. However, dedicating a single memory module or portion of a disc, etc., to the exclusive use of the ECP is one way of separating control processes from user programs in a modular memory system without actual bounds registers. This dedication could include a physically locked external write lockout so that the ECP, once loaded, would be unalterable by programmed action.

Depending on the particular system configuration, the extraction of information from the data processing complex, via a printer or punch, for instance, requires the movement of that information through several of the hierarchical memories. If each memory is protected by the general approach described earlier, then each word of information outputted will be checked several times. Such redundancy may not be necessary in the final analysis, and some of the off-line devices may not need protection, except against misdirected data (by hardware error). A positive lock-up feature for all output devices should be used to hold all peripheral equipment in an unusable state, except when scheduled by the executive control program to perform an input or output function.

Special consideration may be given to tape or similar portable stores, since the data (located on tape reels) may be physically removed from the tape station. The type of protection indicated here may be provided by cryptographic methods. The method chosen need not be very complex, since the tapes are expected to be physically controlled and protected as well. This investigation, however, was considered beyond the scope of this study since it required administrative control.

Associative or content addressable memories, while not too frequently used to date, have considerable appeal in information retrieval applications. They differ from the more familiar random-access memory in that they produce as a minimum a "present-not present" indication, or (at greater complexity) a magnitude comparison, or matching address or addresses (and subsequently the further content) in response to a content-query as applied to the full memory in parallel. This is the reverse of the random-access memory which produces a content in response to the interrogation of a single address. The associative memory is advantageous in that a great deal of time is saved in performing serial searches, since, in effect, all addresses (or data therein contained) are interrogated simultaneously. The principal disadvantage is that it is costly. In general, associative memories may be thought of as a hardware implementation of software since the tasks they perform can be (and now are) performed by the use of a random-access memory and software. Some of the tasks which may be performed by associative memories are: memory protection (by the use of limits or bounds), relative addressing, indexing, sorting, table look-up (thesaurus, etc.), and error correcting. Because of technical and economic factors, it is not now feasible for an associative memory to be used for the entire data base. Now, it is economically questionable even for use as an adjunct to a high-speed memory where it can be used for checking, translating, and accessing. A block of associative memory, if not used to provide protection, may be protected, where necessary, by the previously mentioned checks on the output.

Conventional computers have been organized (following von Neumann's study) so that the internal representation for programmed instructions is a subset of the data representation. This organization provides economy of memory operations with properly functioning equipment and well conceived order codes that are

minimally wasteful of memory space but which also is prone to serious loss of control at the occurrence of malfunction, since unpredictable results occur when data are executed as if they constituted a program.

One alternative organization uses a heterogeneous memory, including a section having an execute-only characteristic, which can be used for the ECP. This will provide assured unalterability of the stored control program.

A second memory section, capable of only being written-in as a result of executing control mode instructions, and contained in the read-only executive program memory, can be used for control tables. These tables are used to describe the multiple user programs, their equipment configuration requirements, and actual equipment assignments made to fulfill these requirements. Prior to releasing the system to a user program, the ECP locks the write control for these control tables. They can be read as required in the execution of the corresponding user program.

A third memory section having unrestricted reading and writing (except for the flag bits) within any word in the assigned range is required for the user programs and corresponding data areas.

Programmed segregation of multiple users from each other is most easily achieved by time sharing the high-speed memory with only one user at a time. The ECP is segregated by the above memory allocation and write lock-outs.

One memory allocation scheme partitions the memory into assignable fixed size groups of words. Readyng a particular user program requires loading into the high-speed memory the program blocks necessary for the user program execution. The IBM System 360* and RCA Spectra 70 have used this approach. This system burdens the compiling program with segmenting to conform to page sizes, often resulting in inefficient use of memory. It simplifies the security protection in that the page boundaries are fixed to one use at a time, and the entire page has the control code of that use. Hardware error is protected by parity check and to some extent, by recognition of misaddressing private blocks.

* An optional write protection feature is announced for some models of the IBM System 360.

Another memory organization, which makes dynamic allocation relatively easy, is the page concept used in the Ferranti-Atlas computer, and more recently, in the GE 635. In the Atlas computer, dynamic allocation allows a main memory of 2^{20} words to be addressable. A page is 512 words of core memory or bulk file sector. Associated with a page in core are an arbitrary high order 11 address bits indicating which page in the bulk file it matches. Thus some small set (as many as the core memory will hold) of the 2^{11} possible blocks of main memory are ready for execution at any time. All addresses are 20 bits. The desired page address for an instruction or data word is determined by comparison with a content-addressable memory which contains the high order 11-bit entry for each page size physical block in main memory and yielding the core page location currently assigned. Absence of the block in core yields return to the supervisor program. The low order 9 bits identify the word in the retrieved page.

An alternative computer organization to the von Neumann organization exists which differentiates data from instructions. One way to achieve this in a homogeneous memory is by the inclusion of one or more flag bits in each memory word. The setting of these bits is under executive control. The condition of these bits may be checked at each access and used not only to differentiate data from program, but also to indicate the type of program (executive or user), or even the security code. The cost of these extra bits and the associated controls is probably only a small part of the cost of the memory system.

When procedure oriented languages (such as those considered most likely for the analysts, input and file management personnel's use) are used to prepare programs, the translator or compiler might be delegated the flag bit setting capability. A machine language program of a user is restricted from using the flag bit setting instructions, since they are privileged instructions.

The combination of memory bounds registers with a fail-safe ECP can be effectively used to restrict or limit a job to the storage locations assigned to it. Storage includes both high-speed (i. e., thin film or core) and low-speed bulk (i. e., disk or tape-type) memories (and perhaps a hierarchy of types). Limiting access to

assigned storage locations includes the following functions:

1. Any input or output between a terminal device and storage;
2. Any transfers between bulk and high-speed memory;
3. Any transfers between parts of the same level of storage;
4. Any manipulation of the instruction location counter or similar device related to program sequencing;
5. Any address alteration outside allowed addresses.

In a multiprogramming environment (without multiprocessing or input/output overlap), the number of pairs of bounds registers available for a high-speed memory determines the number of non-adjacent pieces into which a job to be executed can be separated. Execution from within these pieces is permissible. Job interruption for further ECP action occurs at any attempt to access outside these pieces. Obviously, the minimum ECP interruption occurs if sufficient bounds pairs are used for all non adjacent pieces allocated to a user program. Keeping the number of memory bounds pairs low requires contiguous piece allocation (by occasional relocation of memory contents) and few concurrent users. In the extreme, e.g., Project MAC, the ECP allows exclusive use of one 32-K word unit of high-speed memory to one job at a time with timed priority pre-empting for other users at which time the present contents is stored and a new load is made prior to resuming the new job. This approach has been forced by the absence of hardware specifically designed for time sharing systems. By further complicating this scheme when classified information must be protected, it is necessary to overwrite or erase the entire memory of the prior user before it can be re-allocated, so that no machine-readable sensitive information remains for the successor job to read. As soon as the multiprocessing environment is combined with multiprogramming, the required minimum number of pairs of memory bounds registers increases greatly, unless some means is provided to control the coding of memory bounds registers without requiring an interrupt and ECP action.

The physical make-up of memory bounds registers can assume many forms.

1. The completely hardware form of memory bounds registers presets the bounds limits and interprets those limitations by

means of wired-in logic. This means is equivalent to private memories; it is inflexible and costly.

2. The completely software form of memory bounds registers uses a system program to interpret the effective address of every user program instruction or input/output unit requirement for accessing memory. The interpreted effective address must be found within the user's set of allowable address extremes before remitting the operation to proceed. This is time consuming and does not provide positive protection against hardware malfunction.
3. A combination of the two forms of memory bounds registers is advantageous in that software can set the bounds limits in hardware registers which in turn verify that the effective address of each request for memory access is within the allowable bounds.

When the request does not fall within the prescribed bounds, the hardware signals software (ECP service program) for action. This method has the flexibility required in the multiprogramming, multiprocessing environment since a response appropriate to the need can be programmed to:

1. Reject the request if no access is permitted;
2. Call the required but currently not present (in high-speed memory) data or program;
3. Change the bounds registers to permit the user program access to an authorized area for which no bounds registers were set.

The latter response permits a small set of bounds registers to suffice without unduly constricting the memory allocation algorithm. An extension of this concept, which is the recommended approach, is described with the description of hardware features of the processor in Section III.

APPENDIX IV

ACHIEVING RELIABLE MACHINE OPERATIONS

The problem of constructing a reliable computing circuit from unreliable components was first studied by von Neumann⁽¹⁾ who used neuron-like nets of majority threshold components to produce an overall reliable component. Moore and Shannon⁽²⁾ extended this work by using a sufficiently large number of "crummy" relays in the proper manner to construct arbitrarily good equivalent relays. By replacing actual relays of a relay circuit by equivalent relays, Moore and Shannon noted that the resulting circuit can be arbitrarily good in guarding against pre-specified failure modes. Constant failure rates, independent and intermittent failures, and individual relay performance specified in terms of probability of contacts opening or closing when its coil is energized or not energized were assumed in this study.

Kocher⁽³⁾ showed how the number of redundant relays needed to improve reliability depends on the logical function of the entire circuit. The reliabilities of AND, OR, and EXCLUSIVE-OR relay circuits were considered as a function of the number of relays, the network topology, and the distribution of inputs.

Twenty papers plus an extensive bibliography are included in the proceedings of the 1962 Symposium on "Redundancy Techniques for Computing Systems"⁽⁴⁾. Included is a paper by Kautz⁽⁵⁾ in which efficiency, cost, and optimum codes are

considered for use in digital systems applied at the network or subsystem level. Principal concentration is placed upon parity check codes for detecting or correcting isolated single errors in fixed length blocks of digits.

Massey⁽⁶⁾ discussed applications of error-correcting codes to inter- and intra-computer complex communications and the development of specific techniques suitable for immediate implementation from existing coding techniques.

Armstrong⁽⁷⁾ proposed a general method of applying error correction to a synchronous digital system which, in principle, permits the system to operate continuously even when a fault is present or maintenance is being performed. The redundancy needed to realize the scheme decreases as the system complexity to which it is applied increases. The redundancy is comparable to triplication plus vote taking, which is included as a special case. When the computing circuit is separated into independent sub-units, additional sub-units are added. Boolean functions of these sub-unit inputs form the parity symbols in an error-correcting code, where the outputs of the original sub-units are the information symbols. Ray-Chauduri⁽⁸⁾ developed a coding method leading to minimally-redundant, error-correcting codes and completely solved the problem of constructing minimally-redundant reliable systems whose output is free of error when there is a fault in at most one block of the system. Error correction can imply error detection. Where fail-safe performance is required such that shutdown occurs during repair, the less redundant error-locating codes developed by Wolf and Elspas⁽⁹⁾ can be used with the Armstrong method to locate the single faulty module.

-
1. von Neumann, J., "Probabilistic Logic and the Synthesis of Reliable Organisms from Unreliable Components", Automata Studies, Annals of Math. Studies No. 34, C. E. Shannon and J. McCarthy, Eds., Princeton Univ. Press, 1956, pp. 43-98.
 2. Moore, E. F., and Shannon, C. E., "Reliable Circuits Using Less Reliable Relays", Journal of Franklin Institute 262, 1956, pp. 191-208; 262-297.

3. Kocher, M., "Extension of Moore-Shannon Model for Relay Circuits", IBM Journal of Research and Development, 3 April 1959, pp. 169-188.
4. Wilcox, R. H., and Mann, W. C., Ed. Redundancy Techniques for Computing Systems, Symposium Proceedings, Spartan Books, 1962.
5. Kautz, W. H., "Codes and Coding for Automatic Error Correction Within Digital Systems", In Ref. 4, pp. 152-195.
6. Massey, J. L., "Error Correcting Codes Applied to Computer Technology", Proc. NEC, 19, 1963, pp. 142-147.
7. Armstrong, D. B., "A General Method of Applying Error Correction to Synchronous Digital Systems", Bell System Technical Journal, 40, March 1961, pp. 577-594.
8. Ray-Chauduri, D. K., "On the Construction of Minimally Redundant Reliable System Designs", Bell System Technical Journal, 40, March 1961, pp. 595-612.
9. Wolf, J. K., and Elspas, B., "Error-Locating Codes - A New Concept in Error Control", IEEE Trans. IT-9, April 1963, pp. 124-126.

APPENDIX V

WORK STATION WITH MULTIPLE SIMULTANEOUS USERS

The work station with multiple simultaneous users creates a special security problem. Automatic control of security information release implies machine logging of individuals to whom access to classified information is granted. Partial solutions to the particular difficulties mentioned in Section III are discussed here.

PRIVACY IN USER IDENTIFICATION - AUTHENTICATION

A private means of identification and authentication may exist (such as fingerprints) which is not subject to compromise by others observing its performance. An alternative might be to provide a dual-compartment work station. In the outer compartment, user privacy can be assured so that the identification and authentication may be completed. Upon successful authentication and check that the new user's control profile allows his access to security information currently available in the other compartment, access can be granted to the inner compartment. Exit from the work station would require passing through the outer compartment. The outer door could be unlocked only by performing a suitable user department sequence.

INFORMATION RELEASE RESTRICTION

A work station control profile could be substituted for the individual user's control profile. Such a profile would contain only those control codes common to the control

profile of each user currently authenticated for the work station. A work station control profile, therefore, could not contain more control codes than that user control profile with the least control codes. Since the control codes of requested information are generally unknown to user, and since revelation of the presence of information with a control code not in a control profile would be a security violation, the work station control code approach would restrict the information available to the most broadly authorized user in a work station to a level probably well beneath that to which he would be entitled as a single user. Only in the case where all user control profiles contain the identical control codes is the work station profile not constricting on information release. This case is contrary to the intent of the study.

KEY PATTERN COMPLEXITIES

The work station control profile must be changed every time there is any change in the users present in a work station. Work in process for the work station during such a change might not meet the new conditions of the work station control profile. The output of this work would then have to be deferred. Some technique would also be required for release of deferred work prepared for a group which may not reconvene in the same work station.

The device for generating the work station key pattern would be more complex than for a single user in that it must modify the pattern to identify the particular combinations of users now in the work station. By adding the extra bits necessary to denote these possible combinations, the single user's key pattern could be converted to a work station user's key pattern generator.

CONTROL CODE NAME ASSIGNMENT

The assignment of control code names to a logical record to be added to the data base is the responsibility of the entering user. The set of control code names which one user can assign (as indicated in his user control profile) generally differs from that of any other user. A logical record representing the combined work of a number of users may require a control code name to be assigned which is not available to some participating users. The existence of this control code name and the fact that a particular user can assign it becomes known to other users in the work station. To

alleviate this problem a "pseudo-control code name" could be entered by the user responsible for assignment. A conversion table (protected by a control code available only to that user) could be used internally to the EDP system to provide the actual control code. This requires that each user within the work station be able to initiate private internal processing concurrently with collective processing.

APPENDIX VI
STUDY CONTRIBUTORS

Principal contributors to this study were:

H. W. Bingham	Project Engineer
J. D. Henry	Senior Engineer
Dr. E. Kozik	Project Director
G. E. Lund	Senior Engineer
P. K. Sorensen	Senior Engineering Programmer
J. A. Williams	Senior Engineer
Dr. W. L. Semon	Program Manager

Others who made significant contributions include:

J. M. Ault	Manager, Marketing Technical Staff
G. H. Barnes	Senior Staff Scientist
C. M. Campbell	Senior Engineer
S. A. Cellucci	Engineering Programmer
U. C. S. Dilks	Staff Consultant
W. Goodridge	Security Administrator
J. T. Lynch	Manager, Advanced Development Dept.
J. P. Paris	Marketing Manager
J. A. Petrosky	Associate Engineering Programmer
K. H. Speierman	Senior Staff Scientist

Unclassified
Security Classification

DOCUMENT CONTROL DATA - R&D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) Burroughs Corp. Paoli, Pa.		2a. REPORT SECURITY CLASSIFICATION Unclassified	
		2b. GROUP N/A	
3. REPORT TITLE SECURITY TECHNIQUES FOR EDP OF MULTILEVEL CLASSIFIED INFORMATION			
4. DESCRIPTIVE NOTES (Type of report and inclusive dates) Final Report			
5. AUTHOR(S) (Last name, first name, initial) Bingham, Harvey W.			
6. REPORT DATE December 1965		7a. TOTAL NO. OF PAGES 194	7b. NO. OF REFS 45
8a. CONTRACT OR GRANT NO. AF30(602)-3596		8a. ORIGINATOR'S REPORT NUMBER(S) Burroughs 4424-65-112	
b. PROJECT NO. 4594			
c. Task 459404			
d.		8b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report) RADC-TTR-65-415	
10. AVAILABILITY/LIMITATION NOTICES None			
11. SUPPLEMENTARY NOTES None		12. SPONSORING MILITARY ACTIVITY RADC, Griffiss AFB NY	
13. ABSTRACT The study objective was to develop hardware and software techniques for security (need-to-know) control of on-line users and programmers in multiprogramming, multiprocessing EDP systems of apparent future development. Hardware techniques recommended include: (1) processors having two modes of operation, interrupt entry into control mode in which privileged instructions are executable, flag bits for identification and control of memory words, and address checks against access-differentiated memory bounds; (2) parity checks on intermodule information transfers; (3) input/output control processors which establish and verify peripheral unit connections, check memory addresses against bounds, and confirm security content of record headers being transferred; and (4) bulk file control of physical record integrity, and lock control over write permission and flag bit setting to permit supervisor establishment of control programs. Software techniques reside in the executive control program and are executed in control mode and identified by flag bits. Security routines are described and evaluated which construct, protect, and check access requests against user security control profiles, verify memory bounds and memory blanking, and provide security indicators for input/output. The integrated techniques are applied to control users and system programmers in an advanced modular system. Retrofit of most of the recommended techniques to an existing data processor (the Burroughs D825 modular data processing system) is feasible. An external retrofit unit is described which provides control mode and privileged instructions for single-mode processors.			

Security Classification

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Digital Computers Data Processing System Computers Programming (Computers) Security						

INSTRUCTIONS

1. **ORIGINATING ACTIVITY:** Enter the name and address of the contractor, subcontractor, grantee, Department of Defense activity or other organization (*corporate author*) issuing the report.

2a. **REPORT SECURITY CLASSIFICATION:** Enter the overall security classification of the report. Indicate whether "Restricted Data" is included. Marking is to be in accordance with appropriate security regulations.

2b. **GROUP:** Automatic downgrading is specified in DoD Directive 5200.10 and Armed Forces Industrial Manual. Enter the group number. Also, when applicable, show that optional markings have been used for Group 3 and Group 4 as authorized.

3. **REPORT TITLE:** Enter the complete report title in all capital letters. Titles in all cases should be unclassified. If a meaningful title cannot be selected without classification, show title classification in all capitals in parenthesis immediately following the title.

4. **DESCRIPTIVE NOTES:** If appropriate, enter the type of report, e.g., interim, progress, summary, annual, or final. Give the inclusive dates when a specific reporting period is covered.

5. **AUTHOR(S):** Enter the name(s) of author(s) as shown on or in the report. Enter last name, first name, middle initial. If military, show rank and branch of service. The name of the principal author is an absolute minimum requirement.

6. **REPORT DATE:** Enter the date of the report as day, month, year; or month, year. If more than one date appears on the report, use date of publication.

7a. **TOTAL NUMBER OF PAGES:** The total page count should follow normal pagination procedures, i.e., enter the number of pages containing information.

7b. **NUMBER OF REFERENCES:** Enter the total number of references cited in the report.

8a. **CONTRACT OR GRANT NUMBER:** If appropriate, enter the applicable number of the contract or grant under which the report was written.

8b, 8c, & 8d. **PROJECT NUMBER:** Enter the appropriate military department identification, such as project number, subproject number, system numbers, task number, etc.

9a. **ORIGINATOR'S REPORT NUMBER(S):** Enter the official report number by which the document will be identified and controlled by the originating activity. This number must be unique to this report.

9b. **OTHER REPORT NUMBER(S):** If the report has been assigned any other report numbers (either by the originator or by the sponsor), also enter this number(s).

10. **AVAILABILITY/LIMITATION NOTICES:** Enter any limitations on further dissemination of the report, other than those

imposed by security classification, using standard statements such as:

- (1) "Qualified requesters may obtain copies of this report from DDC."
- (2) "Foreign announcement and dissemination of this report by DDC is not authorized."
- (3) "U. S. Government agencies may obtain copies of this report directly from DDC. Other qualified DDC users shall request through _____."
- (4) "U. S. military agencies may obtain copies of this report directly from DDC. Other qualified users shall request through _____."
- (5) "All distribution of this report is controlled. Qualified DDC users shall request through _____."

If the report has been furnished to the Office of Technical Services, Department of Commerce, for sale to the public, indicate this fact and enter the price, if known.

11. **SUPPLEMENTARY NOTES:** Use for additional explanatory notes.

12. **SPONSORING MILITARY ACTIVITY:** Enter the name of the departmental project office or laboratory sponsoring (paying for) the research and development. Include address.

13. **ABSTRACT:** Enter an abstract giving a brief and factual summary of the document indicative of the report, even though it may also appear elsewhere in the body of the technical report. If additional space is required, a continuation sheet shall be attached.

It is highly desirable that the abstract of classified reports be unclassified. Each paragraph of the abstract shall end with an indication of the military security classification of the information in the paragraph, represented as (TS), (S), (C), or (U).

There is no limitation on the length of the abstract. However, the suggested length is from 150 to 225 words.

14. **KEY WORDS:** Key words are technically meaningful terms or short phrases that characterize a report and may be used as index entries for cataloging the report. Key words must be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location, may be used as key words but will be followed by an indication of technical context. The assignment of links, rules, and weights is optional.